

大数据管理实验代码

大数据2101班 李嘉鹏 U202115652

Lab 1

1.

```
select *  
  
from business  
  
where business_info->'$.city'="Tampa"  
  
order by business_info->'$.review_count' desc  
  
limit 10;
```

2.

```
select  
  
json_keys(business_info) as keys_info,  
json_length(business_info) as key_num_info,  
json_keys(business_info, '$.attributes') as keys_attr,  
json_length(business_info, '$.attributes') as key_num_attr  
  
from business  
  
limit 5;
```

3.

```
select  
  
business_info->'$.name' as name,  
json_type(business_info->'$.name') as name_type,  
business_info->'$.stars' as stars,  
json_type(business_info->'$.stars') as stars_type,  
business_info->'$.attributes' as attributes,  
json_type(business_info->'$. attributes') as attributes_type  
  
from business  
  
limit 5;
```

4.

```
select
```

```
json_unquote(business_info->'$.name') as name,  
business_info->'$.attributes' as attributes,  
business_info->'$.hours' as open_time  
from business  
where business_info->'$.attributes.HasTV'="True"  
and (business_info->'$.hours' is null or business_info->'$.hours.Sunday' is null)  
order by name  
limit 10;
```

5.

```
explain format=json  
select * from user where user_info->'$.name'='Wanda';
```

使用MongoDB查询同样的语句，并使用explain查看查询计划：

```
db.user.find({'name':"Wanda"}).explain("executionStats")
```

6.

```
select json_pretty(business_info)  
from business  
where business_id="4r3Ck65DCG1T6gpWodPyrg";
```

更新操作：

```
update business  
set business_info=json_set(business_info, '$.hours.Tuesday', "16:0-23:0", '$.stars', 4.5, '$.WiFi',  
"Free")  
where business_id="4r3Ck65DCG1T6gpWodPyrg";
```

再次查询：

```
select json_pretty(business_info)  
from business  
where business_id="4r3Ck65DCG1T6gpWodPyrg";
```

7.

```
insert into business(business_id, business_info)
```

```
select "aaaaaabbabbbcccccc2023", business_info from business where business_id='5d-fkQteaqO6CSCqS5q4rw';
```

```
update business
```

```
set business_info=json_remove(business_info, '$.name')
```

```
where business_id="aaaaaabbabbbcccccc2023";
```

查询:

```
select json_pretty(business_info)
```

```
from business
```

```
where business_id="aaaaaabbabbbcccccc2023";
```

8.

```
select
```

```
state,
```

```
json_objectagg(city, count) as city_occ_num
```

```
from
```

```
(
```

```
select
```

```
business_info->'$.state' as state,
```

```
json_unquote(business_info->'$.city') as city,
```

```
count(*) as count
```

```
from business
```

```
group by state, city
```

```
) as sub
```

```
group by state
```

```
order by state;
```

9.

```
select sub.userid as user_id,
```

```
uu.user_info->'$.name' as name,
```

```

sub.textarray as text_array from
(
select t.user_id as userid,
json_arrayagg(t.tip_info->'$.text') as textarray
from tip t
join user u on REGEXP_LIKE(u.user_info->'$.friends', t.user_id)
where u.user_id='__1cb6cwl3uAbMTK3xaGbg'
group by t.user_id
) as sub, user uu
where uu.user_id = sub.userid
order by name;

```

10.

```

select
a.business_info->'$.name' as name1,
a.business_info->'$.city' as city1,
b.business_info->'$.name' as name2,
b.business_info->'$.city' as city2,
a.business_info->'$.hours' as hours1,
b.business_info->'$.hours' as hours2,
JSON_OVERLAPS(a.business_info->'$.hours', b.business_info->'$.hours') as has_same_opentime
from business a
join business b
on a.business_info->'$.city' = 'Edmonton' and b.business_info->'$.city' = 'Elsmere';

```

11.

```

select
user_info->'$.name' as name,
user_info->'$.average_stars' as avg_stars,
JSON_ARRAY(user_info->'$.funny', user_info->'$.useful', user_info->'$.cool', user_info->'$.funny'+user_info->'$.useful'+user_info->'$.cool') as '[funny,useful,cool,sum]'
from user
where user_info->'$.funny'>2000
and user_info->'$.average_stars'>4.0

```

order by user_info->'\$.average_stars' desc

limit 10;

报错:

手动修改缓存池大小即可

在前面加上explain format=json

和第一题对比:

12.

select

json_merge_preserve(b.business_info, u.user_info)

from

(

select business_id,

count(*) as count

from tip

group by business_id

order by count desc

limit 1

) as tb

join

(

select user_id,

count(*) as count

from tip

group by user_id

order by count desc

limit 1

) as tu

join business b on tb.business_id=b.business_id

join user u on tu.user_id=u.user_id;

13.

select

sub.business_name,

sub.review_count as business_review_count,

case when sub.hours->'\$.Tuesday' is null then 0 else 1 end as business_open_on_Tuesday,

jt.time_slot

from

(select

business_info->'\$.name' as business_name,

business_info->'\$.review_count' as review_count,

business_info->'\$.hours' as hours

from business

order by business_info->'\$.review_count' desc

limit 3) as sub

join json_table(

sub.hours,

"\$.*" columns (

time_slot VARCHAR(255) PATH '\$')

) as jt on 1=1

order by sub.business_name;

Lab 2

1.

db.review.find().limit(2).skip(6)

```
test1 - root@ecs-c925: ~/data - Xshell 7
文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W) 帮助(H)
ssh://root:*****@1.94.55.43:22
要添加当前会话, 点击左侧的箭头按钮。
会话管理器
1 test1
所有会话
test1
名称 test1
主机 1.94.55.43
端口 22
协议 SSH
用户名 root
说明
> db.review.find().limit(2).skip(6)
{ "_id" : ObjectId("600d7ea4f5e9bd91d7c3001e"), "review_id" : "G7XHMxG0bX9oBJNEC64IFg", "u
ser_id" : "jlu4CztcsXrKx56ba1a5AQ", "business_id" : "3fw2X5bZYew9xCz_zGh0Hg", "stars" : 3,
"useful" : 5, "funny" : 4, "cool" : 5, "text" : "Tracy dessert had a big name in Hong Kon
g and the one in First Markham place has been here for many years now! \n\nCame in for som
e Chinese dessert, and I must say their selection has increased tremendously over the year
s. I might as well add that the price has also increased tremendously as well. The waitres
s gave us tea, which I could taste had red date in it. Fancy!\n\nA simple taro with coconu
t with tapioca pearls was like $5.25 or something. Basically all the desserts were more th
an $5. That's crazy! I can literally just make this dessert at home and for a bowl, it wou
ld probably cost like $0.50. A few years ago, I think I can still get it for like $3-$4, w
hich is more reasonable, but wow, more than $5 is a little over the top for this dessert.
Though I must say, it is Tracy Dessert, and they are a little more on the expensive side.
\n\nI also saw other items on the menu like fish balls, chicken wings, shaved ice. My frie
nd got a mango drink with fresh mango in it! \n\nI'm also surprised how many people come t
o Tracy Dessert after work. We came on a Sunday and the tables were always filled. I think
the amount of tables they had were just perfect because no one really waited for seats fo
r a long time, but the tables kept filling up once a table was finished.", "date" : "2016-
05-07 01:21:02" }
{ "_id" : ObjectId("600d7ea4f5e9bd91d7c3001f"), "review_id" : "8e9HxxLjjqC9ez5ezZn7iQ", "u
ser_id" : "d6xvYpyzcfbF_AZ8vMB7QA", "business_id" : "zv0-PJcPnK4fgAVUnEXYAA", "stars" : 1,
"useful" : 3, "funny" : 1, "cool" : 1, "text" : "This place has gone down hill. Clearly
they have cut back on staff and food quality\n\nMany of the reviews were written before th
e menu changed. I've been going for years and the food quality has gone down hill.\n\nThe
service is slow & my salad, which was $15, was as bad as it gets.\n\nIt's just not worth
spending the money on this place when there are so many other options.", "date" : "2010-10
-05 19:12:35" }
>
ssh://root@1.94.55.43:22 SSH2 xterm 90x28 25.3 1 会话 CAP NUM
```

2.

db.business.find({'city':'Las Vegas'}).limit(5)

```
> db.business.find({'city':'Las Vegas'}).limit(5)
{ "_id" : ObjectId("6016cb64af81085b0f2183c8"), "business_id" : "gbQ07vr-caG_AlugSmGhWg", "name" : "Supercuts", "address" : "4545 E Tropicana Rd Ste 8, Tropicana", "city" : "Las Vegas", "state" : "NV", "postal_code" : "89121", "latitude" : 36.099872, "longitude" : -115.074574, "stars" : 3.5, "review_count" : 3, "is_open" : 1, "attributes" : { "RestaurantsPriceRange2" : "3", "GoodForKids" : "True", "BusinessAcceptsCreditCards" : "True", "ByAppointmentOnly" : "False", "BikeParking" : "False", "Categories" : [ "Hair Salons", "Hair Stylists", "Barbers", "Men's Hair Salons", "Cosmetics & Beauty Supply", "Shopping", "Beauty & Spas" ], "hours" : { "Monday" : "10:0-19:0", "Tuesday" : "10:0-19:0", "Wednesday" : "10:0-19:0", "Thursday" : "10:0-19:0", "Friday" : "10:0-19:0", "Saturday" : "10:0-19:0", "Sunday" : "10:0-18:0" }, "loc" : { "type" : "Point", "coordinates" : [ -115.074574, 36.099872 ] } }
{ "_id" : ObjectId("6016cb64af81085b0f2183d2"), "business_id" : "P2-LZs5lhSeutkQYU8pFg", "name" : "Carluccio's Tivoli Gardens", "address" : "1775 E Tropicana Ave, Ste 29", "city" : "Las Vegas", "state" : "NV", "postal_code" : "89119", "latitude" : 36.1000163, "longitude" : -115.1285285, "stars" : 4, "review_count" : 40, "is_open" : 0, "attributes" : { "OutdoorSeating" : "False", "BusinessAcceptsCreditCards" : "True", "RestaurantsDelivery" : "False", "RestaurantsReservations" : "True", "RestaurantsAttire" : "casual", "Ambience" : { "romantic" : "True", "intimate" : "False", "hipster" : "False", "diver" : "False", "classy" : "True", "trendy" : "False", "upscale" : "False", "casual" : "False", "HasTV" : "False", "BYOB/Corkage" : "no", "NoiseLevel" : "u'quiet", "RestaurantsTakeOut" : "True", "RestaurantsPriceRange2" : "2", "RestaurantsGoodForGroups" : "True", "WiFi" : "u'no", "Caters" : "True", "GoodForKids" : "True", "Alcohol" : "u'full_bar", "BusinessParking" : { "garage" : "False", "street" : "False", "validated" : "False", "lot" : "True", "valet" : "False" }, "categories" : [ "Restaurants", "Italian" ], "hours" : null, "loc" : { "type" : "Point", "coordinates" : [ -115.1285285, 36.1000163 ] } }
{ "_id" : ObjectId("6016cb64af81085b0f2183d3"), "business_id" : "mh-K016QAOXWqZ08PHF60", "name" : "Myron Hensel Photography", "address" : "", "city" : "Las Vegas", "state" : "NV", "postal_code" : "89121", "latitude" : 36.1165487, "longitude" : -115.0881146, "stars" : 5, "review_count" : 21, "is_open" : 1, "attributes" : { "BusinessAcceptsCreditCards" : "True" }, "categories" : [ "Event Planning & Services", "Photographers", "Professional Services" ], "hours" : { "Monday" : "0:0-0:0", "Tuesday" : "0:0-0:0", "Wednesday" : "0:0-0:0", "Thursday" : "0:0-0:0", "Friday" : "0:0-0:0", "Saturday" : "0:0-0:0", "Sunday" : "0:0-0:0" }, "loc" : { "type" : "Point", "coordinates" : [ -115.0881146, 36.1165487 ] } }
{ "_id" : ObjectId("6016cb64af81085b0f2183d5"), "business_id" : "dPMxHgyTY6F873843dHAA", "name" : "Fremont Arcade", "address" : "450 Fremont St, Ste 179", "city" : "Las Vegas", "state" : "NV", "postal_code" : "89101", "latitude" : 36.169993, "longitude" : -115.140685, "stars" : 4.5, "review_count" : 38, "is_open" : 1, "attributes" : { "GoodForKids" : "True" }, "categories" : [ "Arcades", "Arts & Entertainment" ], "hours" : { "Monday" : "11:0-0:0", "Tuesday" : "11:0-0:0", "Wednesday" : "11:0-0:0", "Thursday" : "11:0-0:0", "Friday" : "11:0-1:0", "Saturday" : "11:0-1:0", "Sunday" : "11:0-0:0" }, "loc" : { "type" : "Point", "coordinates" : [ -115.140685, 36.169993 ] } }
{ "_id" : ObjectId("6016cb64af81085b0f2183d7"), "business_id" : "KWywu2TETPMWR9JnBc0MYQ", "name" : "Hunk Mansion", "address" : "6007 Dean Martin Dr", "city" : "Las Vegas", "state" : "NV", "postal_code" : "89118", "latitude" : 36.0801670, "longitude" : -115.1827558, "stars" : 4, "review_count" : 107, "is_open" : 1, "attributes" : { "BikeParking" : "False", "Ambience" : { "romantic" : "False", "intimate" : "True", "hipster" : "False", "diver" : "False", "classy" : "True", "trendy" : "False", "upscale" : "False", "casual" : "False", "RestaurantsPriceRange2" : "2", "HasTV" : "True", "Music" : { "dj" : "True", "background_music" : "False", "no_music" : "False", "jukebox" : "False", "live" : "False", "video" : "False", "karaoke" : "False", "RestaurantsGoodForGroups" : "True", "OutdoorSeating" : "True", "RestaurantsReservations" : "True", "BusinessParking" : { "garage" : "False", "street" : "False", "validated" : "False", "lot" : "True", "valet" : "True", "GoodForKids" : "False", "Alcohol" : "u'full_bar", "NoiseLevel" : "loud", "BusinessAcceptsCreditCards" : "True", "GoodForDancing" : "True", "categories" : [ "Nightlife", "Arts & Entertainment", "Bars", "Strip Clubs", "Adult Entertainment", "Dance Clubs" ], "hours" : { "Thursday" : "19:30-2:0", "Friday" : "19:30-3:0", "Saturday" : "19:30-3:0", "Sunday" : "19:30-2:0" }, "loc" : { "type" : "Point", "coordinates" : [ -115.1827558, 36.0801670 ] } } }
```

3.

db.user.find({'name':'Steve'},{'useful':1,'cool':1}).limit(10)

```
> db.user.find({'name':'Steve'},{'useful':1,'cool':1}).limit(10)
{ "_id" : ObjectId("600d35c72cb85d6290190f0c"), "useful" : 680, "cool" : 383 }
{ "_id" : ObjectId("600d35c72cb85d62901911d1"), "useful" : 33, "cool" : 10 }
{ "_id" : ObjectId("600d35c72cb85d629019141f"), "useful" : 4359, "cool" : 3663 }
{ "_id" : ObjectId("600d35c72cb85d62901914b4"), "useful" : 4, "cool" : 2 }
{ "_id" : ObjectId("600d35c72cb85d6290191803"), "useful" : 31343, "cool" : 28316 }
{ "_id" : ObjectId("600d35c72cb85d6290191a86"), "useful" : 2, "cool" : 2 }
{ "_id" : ObjectId("600d35c72cb85d6290191ad0"), "useful" : 9, "cool" : 0 }
{ "_id" : ObjectId("600d35c72cb85d6290191b18"), "useful" : 709, "cool" : 339 }
{ "_id" : ObjectId("600d35c72cb85d6290191b88"), "useful" : 295, "cool" : 115 }
{ "_id" : ObjectId("600d35c72cb85d6290191bb0"), "useful" : 41, "cool" : 8 }
```

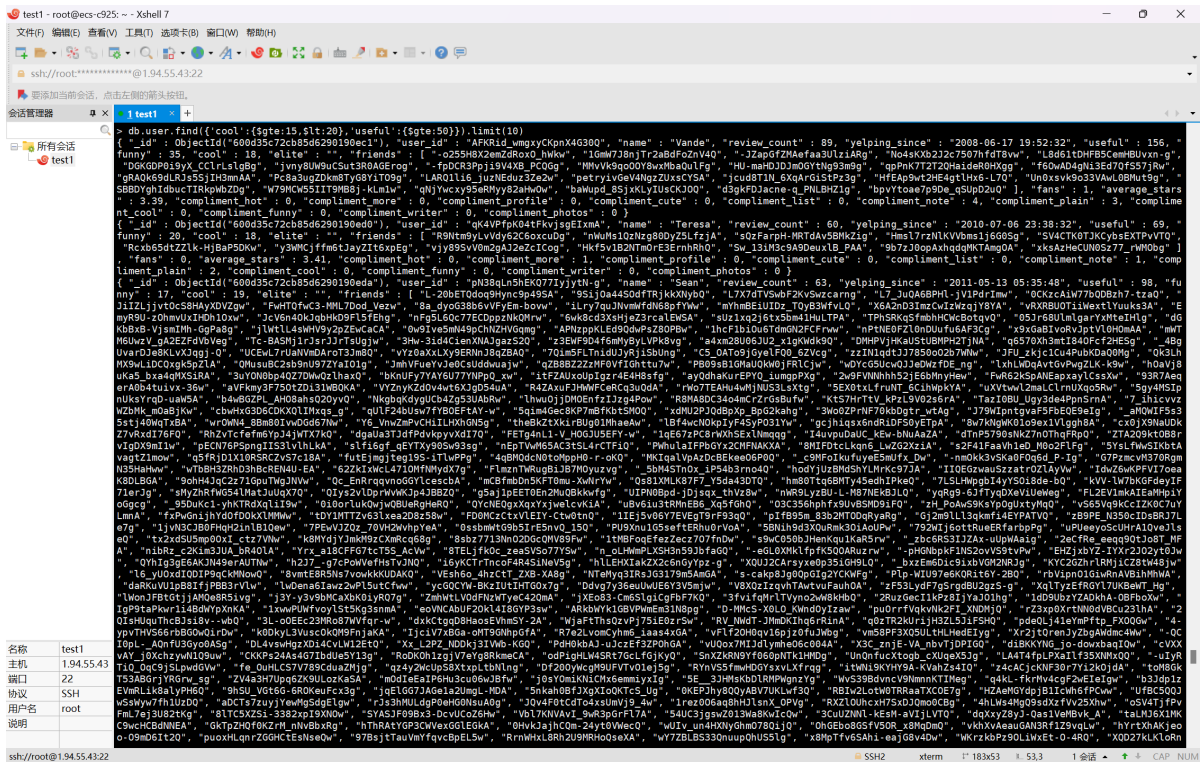
4.

db.user.find({'funny':{\$in:[66,67,68]}},{'name':1,'funny':1}).limit(20)

```
> db.user.find({'funny':{'$in:[66,67,68]}},{ 'name':1,'funny':1}).limit(20)
{ "_id" : ObjectId("600d35c72cb85d62901910ac"), "name" : "Aileen", "funny" : 68 }
{ "_id" : ObjectId("600d35c72cb85d629019119c"), "name" : "Alison", "funny" : 67 }
{ "_id" : ObjectId("600d35c72cb85d62901911b6"), "name" : "Karin", "funny" : 66 }
{ "_id" : ObjectId("600d35c72cb85d6290191290"), "name" : "AshLee", "funny" : 68 }
{ "_id" : ObjectId("600d35c72cb85d62901913ea"), "name" : "Angie", "funny" : 66 }
{ "_id" : ObjectId("600d35c72cb85d62901914b6"), "name" : "Kristen", "funny" : 66 }
{ "_id" : ObjectId("600d35c72cb85d6290191840"), "name" : "Kathleen", "funny" : 67 }
{ "_id" : ObjectId("600d35c72cb85d62901918f3"), "name" : "Ted", "funny" : 67 }
{ "_id" : ObjectId("600d35c72cb85d6290191907"), "name" : "Donald", "funny" : 66 }
{ "_id" : ObjectId("600d35c72cb85d6290191911"), "name" : "Heidi", "funny" : 67 }
{ "_id" : ObjectId("600d35c72cb85d6290191953"), "name" : "Angelica", "funny" : 66 }
{ "_id" : ObjectId("600d35c72cb85d62901919da"), "name" : "Julia", "funny" : 68 }
{ "_id" : ObjectId("600d35c72cb85d6290191a2e"), "name" : "Melinda", "funny" : 68 }
{ "_id" : ObjectId("600d35c72cb85d6290191b03"), "name" : "Lisa", "funny" : 68 }
{ "_id" : ObjectId("600d35c72cb85d6290191b45"), "name" : "Sofia", "funny" : 68 }
{ "_id" : ObjectId("600d35c72cb85d6290191bbd"), "name" : "Amy", "funny" : 68 }
{ "_id" : ObjectId("600d35c72cb85d6290191c77"), "name" : "Keane", "funny" : 67 }
{ "_id" : ObjectId("600d35c72cb85d6290191ce2"), "name" : "Karina", "funny" : 66 }
{ "_id" : ObjectId("600d35c72cb85d6290191f29"), "name" : "Jess", "funny" : 66 }
{ "_id" : ObjectId("600d35c72cb85d629019203d"), "name" : "Linda", "funny" : 68 }
```

5.

db.user.find({'cool':{'\$gte:15,\$lte:20},'useful':{'\$gte:50}}).limit(10)



6.

db.business.aggregate({'\$group':{'_id':"total":{"\$sum:1}}})

```
> db.business.aggregate({'$group':{'_id':"total":{"$sum:1}}})
{ "_id" : "", "total" : 192609 }
```

db.business.explain("executionStats").count()


```

> db.business.explain("executionStats").count()
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "yelp.business",
    "indexFilterSet" : false,
    "winningPlan" : {
      "stage" : "RECORD_STORE_FAST_COUNT"
    },
    "rejectedPlans" : [ ]
  },
  "executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 0,
    "executionTimeMillis" : 0,
    "totalKeysExamined" : 0,
    "totalDocsExamined" : 0,
    "executionStages" : {
      "stage" : "RECORD_STORE_FAST_COUNT",
      "nReturned" : 0,
      "executionTimeMillisEstimate" : 0,
      "works" : 1,
      "advanced" : 0,
      "needTime" : 0,
      "needYield" : 0,
      "saveState" : 0,
      "restoreState" : 0,
      "isEOF" : 1,
      "nCounted" : 192609,
      "nSkipped" : 0
    }
  },
  "serverInfo" : {
    "host" : "ecs-c925",
    "port" : 27017,
    "version" : "4.4.17",
    "gitVersion" : "85de0cc83f4dc64dbbac7fe028a4866228c1b5d1"
  },
  "ok" : 1
}

```

7.

```
db.business.find({'city': {'$in': ['Westlake', 'Calgary']}})
```

```
test1 - root@ecs-c25: ~ - Xshell 7
文件(F) 编辑(E) 查看(V) 工具(T) 选项卡(B) 窗口(W) 帮助(H)
ssh://root@194.55.43.22
要添加当前会话，点击左方的箭头按钮。
会话管理器
所有会话
test1
> db.business.find({'city':{'$in':['Westlake','Calgary']}})
{"_id": "ObjectID('6016c6b4af81085b0f2183c7')", "business_id": "53ucpCfHtJh5r1JabJdG", "name": "Edgeworx Studio", "address": "20 Douglas Woods Drive Southeast", "city": "Calgary", "state": "AB", "postal_code": "T2Z 1K4", "latitude": 50.9436456, "longitude": -114.0018283, "stars": 3.5, "review_count": 7, "is_open": 1, "attributes": { "RestaurantsPriceRange2": "2", "BusinessParking": { "garage": False, "street": "False", "validated": False, "lot": True, "valet": False }, "ByAppointmentOnly": "True" }, "categories": [ "Beauty & Spas", "Hair Salons" ], "hours": null, "loc": { "type": "Point", "coordinates": [ -114.0018283, 50.9436456 ] } }
{"_id": "ObjectID('6016c6b4af81085b0f2183cb')", "business_id": "fc0X0Z2qVzqG03G10Xmg", "name": "Nucleus Information Service", "address": "1210 8th Street SW, Unit 220", "city": "Calgary", "state": "AB", "postal_code": "T2R 1L3", "latitude": 51.0417711, "longitude": -114.081109, "stars": 2, "review_count": 5, "is_open": 1, "attributes": { "BikeParking": "False", "ByAppointmentOnly": "False", "BusinessParking": { "garage": False, "street": "True", "validated": False, "lot": False, "valet": False }, "RestaurantsPriceRange2": "1" }, "categories": [ "Local Services", "Professional Services", "Computers", "Shopping", "Home Services", "IT Services & Computer Repair", "Internet Service Providers", "Web Design" ], "hours": { "Monday": "17:0", "Tuesday": "9:0-17:0", "Wednesday": "9:0-17:0", "Thursday": "9:0-17:0", "Friday": "9:0-17:0" }, "loc": { "type": "Point", "coordinates": [ -114.081109, 51.0417711 ] } }
{"_id": "ObjectID('6016c6b4af81085b0f2183e9')", "business_id": "MsRvdPnuv6QulN5Vxrw", "name": "Brian's Furniture", "address": "30808 Center Ridge Rd", "city": "Westlake", "state": "ON", "postal_code": "M4I4S", "latitude": 41.4227938, "longitude": -81.9622151, "stars": 5, "review_count": 3, "is_open": 1, "attributes": { "Shopping": "Home Services", "Interior Design": "Rugs", "Furniture Stores", "Home & Garden", "Home Decor", "Mattresses" }, "hours": { "Monday": "11:0-20:0", "Tuesday": "10:0-20:0", "Wednesday": "10:0-20:0", "Thursday": "10:0-18:0", "Friday": "10:0-18:0", "Saturday": "10:0-18:0", "Sunday": "10:0-17:0" }, "loc": { "type": "Point", "coordinates": [ -81.9622151, 41.4227938 ] } }
{"_id": "ObjectID('6016c6b4af81085b0f2183ee')", "business_id": "e.EHySpQmVLvZfdmReAo", "name": "Pearl Garden", "address": "2925 32 Avenue NE", "city": "Calgary", "state": "AB", "postal_code": "T1Y1", "latitude": 51.0811622, "longitude": -113.9934738, "stars": 2, "review_count": 4, "is_open": 0, "attributes": { "BusinessParking": { "garage": False, "street": "False", "validated": False, "lot": False, "valet": False }, "RestaurantsTakeOut": "True", "RestaurantsGoodForGroups": "True", "OutdoorSeating": "False", "RestaurantsPriceRange2": "1", "GoodForkids": "True", "RestaurantsDelivery": "False", "RestaurantsTakeOut": "True", "RestaurantsAttire": "u'casual'", "categories": [ "Chinese", "Dim Sum", "Restaurants" ], "hours": null, "loc": { "type": "Point", "coordinates": [ -113.9934738, 51.0811622 ] } } }
{"_id": "ObjectID('6016c6b4af81085b0f2183f1')", "business_id": "Jff5hTK1ZMLK506sOSWdW", "name": "Radisson Hotel & Conference Centre Calgary Airport", "address": "6620 36 Street NE", "city": "Calgary", "state": "AB", "postal_code": "T3J 4C8", "latitude": 51.118915249, "longitude": -113.9804801532, "stars": 3.5, "review_count": 14, "is_open": 1, "attributes": { "RestaurantsPriceRange2": "2", "WiFi": "free" }, "categories": [ "Hotels & Travel", "Hotels", "Event Planning & Services" ], "hours": null, "loc": { "type": "Point", "coordinates": [ -113.9804801532, 51.118915249 ] } } }
{"_id": "ObjectID('6016c6b4af81085b0f2183fa')", "business_id": "c08FKduGfvsKv11NZKsAg", "name": "Tomkins Park", "address": "880 17 Avenue SW", "city": "Calgary", "state": "AB", "postal_code": "T2T 0A2", "latitude": 51.0379943848, "longitude": -114.0813674927, "stars": 3.5, "review_count": 3, "is_open": 1, "attributes": { "GoodForkids": "True", "BusinessParking": { "garage": False, "street": "False", "validated": False, "lot": False, "valet": False }, "BikeParking": "True", "HasTV": "True", "OutdoorSeating": "False", "RestaurantsReservations": "True", "categories": [ "type": "Point", "coordinates": [ -114.0813674927, 51.0379943848 ] } } }
{"_id": "ObjectID('6016c6b4af81085b0f218411')", "business_id": "PKDghu4aan2_wxrhXjTEg", "name": "Mirakuru", "address": "529-17th Avenue SW", "city": "Calgary", "state": "AB", "postal_code": "T2S 0A9", "latitude": 51.037777046, "longitude": -114.0733513906, "stars": 3.5, "review_count": 16, "is_open": 0, "attributes": { "WiFi": "free", "GoodForDancing": "False", "RestaurantsPriceRange2": "2", "RestaurantsAttire": "u'casual'", "BikeParking": "True", "HasTV": "True", "OutdoorSeating": "False", "RestaurantsReservations": "True", "categories": [ "type": "Point", "coordinates": [ -114.0733513906, 51.037777046 ] } } }
{"_id": "ObjectID('6016c6b4af81085b0f218424')", "business_id": "p3d5QmQHBer3UXG3ZvW", "name": "Carl's Jr.", "address": "309 30 Avenue SW", "city": "Calgary", "state": "AB", "postal_code": "T2S 1H3", "latitude": 51.0084601753, "longitude": -114.0687542984, "stars": 3.5, "review_count": 6, "is_open": 1, "attributes": { "WiFi": "free", "Caters": "False", "HasTV": "True", "RestaurantsDelivery": "False", "RestaurantsAttire": "casual", "RestaurantsPriceRange2": "2", "Alcohol": "u'none'", "NoiseLevel": "u'average'", "Ambience": "touristy", "classy": "hipster", "romantic": "False", "intimate": "False", "trendy": "False", "upscale": "False", "casual": "True", "BikeParking": "True", "DriveThru": "True", "GoodForMeal": { "dessert": "False", "latenight": "False", "lunch": "True", "dinner": "True", "brunch": "False", "breakfast": "False", "RestaurantsTakeOut": "True", "OutdoorSeating": "False", "RestaurantsTableService": "True", "RestaurantsReservations": "False", "BusinessParking": { "garage": False, "street": "False", "validated": False, "lot": True, "valet": False }, "RestaurantsGoodForGroups": "True", "GoodForkids": "True" }, "categories": [ "American (Traditional)", "Breakfast & Brunch", "Restaurants", "Burgers", "Fast Food" ], "hours": { "Monday": "10:30-22:0", "Tuesday": "10:30-22:0", "Wednesday": "10:30-22:0", "Thursday": "10:30-22:0", "Friday": "10:30-22:0", "Saturday": "11:0-23:0", "Sunday": "11:0-23:0" }, "loc": { "type": "Point", "coordinates": [ -114.0687542984, 51.0084601753 ] } } }
{"_id": "ObjectID('6016c6b4af81085b0f218490')", "business_id": "agdyRoBcNc29_j2vJmKW", "name": "Phil's Restaurants", "address": "3210 17 Avenue SE", "city": "Calgary", "state": "AB", "postal_code": "T2A 0P9", "latitude": 51.0381747, "longitude": -113.9892355, "stars": 3.5, "review_count": 3, "is_open": 1, "attributes": { "Caters": "False", "HasTV": "False", "OutdoorSeating": "True", "RestaurantsGoodForGroups": "True", "GoodForkids": "True", "WiFi": "u'no'", "RestaurantsDelivery": "False", "RestaurantsReservations": "False", "Alcohol": "u'beer_and_wine'", "RestaurantsTakeOut": "True", "NoiseLevel": "u'average'", "Ambience": { "romantic": "False", "intimate": "False", "classy": "False", "hipster": "False", "trendy": "False", "upscale": "False", "casual": "False", "RestaurantsAttire": "u'casual'", "BusinessParking": { "garage": "False", "street": "False", "validated": "False", "lot": "True", "valet": "False" }, "RestaurantsPriceRange2": "2", "BikeParking": "False" }, "categories": [ "Breakfast & Brunch", "Diners", "Restaurants", "Coffee & Tea" ] } }
```

名称	test1
主机	194.55.43.22
端口	22
协议	SSH
用户名	root
说明	

```
> db.business.find({'city':{'$in':['Westlake','Calgary']}}).count()
8125
```

8.

db.business.find({'categories':{'\$size:6}},{'_id':1,'categories':1}).limit(10)

```
> db.business.find({'categories':{'$size:6}},{'_id':1,'categories':1}).limit(10)
{"_id": "ObjectID('6016c6b4af81085b0f2183cc')", "categories": [ "Restaurants", "Breakfast & Brunch", "Mexican", "Tacos", "Tex-Mex", "Fast Food" ] }
{"_id": "ObjectID('6016c6b4af81085b0f2183cd')", "categories": [ "Bars", "Nightlife", "Pubs", "Barbers", "Beauty & Spas", "Irish Pub" ] }
{"_id": "ObjectID('6016c6b4af81085b0f2183d4')", "categories": [ "Trainers", "Health & Medical", "Active Life", "Physical Therapy", "Gyms", "Fitness & Instruction" ] }
{"_id": "ObjectID('6016c6b4af81085b0f2183d7')", "categories": [ "Nightlife", "Arts & Entertainment", "Bars", "Strip Clubs", "Adult Entertainment", "Dance Clubs" ] }
{"_id": "ObjectID('6016c6b4af81085b0f2183da')", "categories": [ "Mexican", "Restaurants", "Patisserie/Cake Shop", "Food", "Bars", "Nightlife" ] }
{"_id": "ObjectID('6016c6b4af81085b0f2183e8')", "categories": [ "Auto Glass Services", "Auto Detailing", "Automotive", "Auto Parts & Supplies", "Auto Customization", "Vehicle Wraps" ] }
{"_id": "ObjectID('6016c6b4af81085b0f2183f3')", "categories": [ "Restaurants", "Breakfast & Brunch", "Bars", "Modern European", "Nightlife", "Wine Bars" ] }
{"_id": "ObjectID('6016c6b4af81085b0f218400')", "categories": [ "Men's Clothing", "Sporting Goods", "Shopping", "Fashion", "Women's Clothing", "Sports Wear" ] }
{"_id": "ObjectID('6016c6b4af81085b0f218403')", "categories": [ "Nightlife", "Bars", "Polish", "Modern European", "Restaurants", "Vegan" ] }
{"_id": "ObjectID('6016c6b4af81085b0f218411')", "categories": [ "Nightlife", "Italian", "Restaurants", "Japanese", "Lounges", "Bars" ] }
```

9.

db.business.find({'business_id': '5JucpCfHzlth5r1JabJdG' }).explain("executionStats")

```
> db.business.find({ business_id: "5JucpCfHZltJh5r1JabjDg" }).explain("executionStats");
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "yelp.business",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "business_id" : {
        "$eq" : "5JucpCfHZltJh5r1JabjDg"
      }
    },
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "business_id" : {
          "$eq" : "5JucpCfHZltJh5r1JabjDg"
        }
      },
      "direction" : "forward"
    },
    "rejectedPlans" : [ ]
  },
  "executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 1,
    "executionTimeMillis" : 64,
    "totalKeysExamined" : 0,
    "totalDocsExamined" : 192609,
    "executionStages" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "business_id" : {
          "$eq" : "5JucpCfHZltJh5r1JabjDg"
        }
      },
      "nReturned" : 1,
      "executionTimeMillisEstimate" : 6,
      "works" : 192611,
      "advanced" : 1,
      "needTime" : 192609,
      "needYield" : 0,
      "saveState" : 192,
      "restoreState" : 192,
      "isEOF" : 1,
      "direction" : "forward",
      "docsExamined" : 192609
    }
  },
  "serverInfo" : {
    "host" : "ecs-c925",
    "port" : 27017,
    "version" : "4.4.17",
    "gitVersion" : "85de0cc83f4dc64dbbac7fe028a4866228c1b5d1"
  }
}
```

可以看出使用了COLLSCAN方式（遍历整个集合）

执行计划关注parsedQuery（解析的查询表达式）、winningPlan（查询执行策略）、rejectedPlans（考虑但未选中的执行策略）

通过物理方式进行优化：为business_id建立索引。

```
db.business.createIndex({'business_id':1})
```

```
> db.business.createIndex({'business_id':1})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```

再次使用explain查看执行过程:

```
> db.business.find({ business_id: "5JucpCfHZltJh5r1JabjDg" }).explain("executionStats")
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "yelp.business",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "business_id" : {
        "$eq" : "5JucpCfHZltJh5r1JabjDg"
      }
    },
    "winningPlan" : {
      "stage" : "FETCH",
      "inputStage" : {
        "stage" : "IXSCAN",
        "keyPattern" : {
          "business_id" : 1
        },
        "indexName" : "business_id_1",
        "isMultiKey" : false,
        "multiKeyPaths" : {
          "business_id" : [ ]
        },
        "isUnique" : false,
        "isSparse" : false,
        "isPartial" : false,
        "indexVersion" : 2,
        "direction" : "forward",
        "indexBounds" : {
          "business_id" : [
            ["5JucpCfHZltJh5r1JabjDg\","5JucpCfHZltJh5r1JabjDg\"]
          ]
        }
      },
      "rejectedPlans" : [ ]
    },
    "executionStats" : {
      "executionSuccess" : true,
      "nReturned" : 1,
      "executionTimeMillis" : 0,
      "totalKeysExamined" : 1,
      "totalDocsExamined" : 1,
      "executionStages" : {
        "stage" : "FETCH",
        "nReturned" : 1,
        "executionTimeMillisEstimate" : 0,
        "works" : 2,
        "advanced" : 1,
        "needTime" : 0,
        "needYield" : 0,
        "saveState" : 0,
        "restoreState" : 0,
```

```

        "isEOF" : 1,
        "docsExamined" : 1,
        "alreadyHasObj" : 0,
        "inputStage" : {
            "stage" : "IXSCAN",
            "nReturned" : 1,
            "executionTimeMillisEstimate" : 0,
            "works" : 2,
            "advanced" : 1,
            "needTime" : 0,
            "needYield" : 0,
            "saveState" : 0,
            "restoreState" : 0,
            "isEOF" : 1,
            "keyPattern" : {
                "business_id" : 1
            },
            "indexName" : "business_id_1",
            "isMultiKey" : false,
            "multiKeyPaths" : {
                "business_id" : [ ]
            },
            "isUnique" : false,
            "isSparse" : false,
            "isPartial" : false,
            "indexVersion" : 2,
            "direction" : "forward",
            "indexBounds" : {
                "business_id" : [
                    ["\5JucpCfHZltJh5r1JabjDg\","5JucpCfHZltJh5r1JabjDg\"]
                ]
            },
            "keysExamined" : 1,
            "seeks" : 1,
            "dupsTested" : 0,
            "dupsDropped" : 0
        }
    },
    "serverInfo" : {
        "host" : "ecs-c925",
        "port" : 27017,
        "version" : "4.4.17",
        "gitVersion" : "85de0cc83f4dc64dbbac7fe028a4866228c1b5d1"
    },
    "ok" : 1
}

```

很明显可以看到时间变短 (64ms->0ms) , 且查询方式变为FETCH, 索引名为business_id_1

10.

```

db.business.aggregate([

{$group:{cnt:{$sum:1}, '_id': '$stars'}},

{$sort:{'_id': -1}}])

```

```

> db.business.aggregate([ {$group:{cnt:{$sum:1}, '_id': '$stars'}}, {$sort:{'_id': -1}}])
{ "_id" : 5, "cnt" : 28216 }
{ "_id" : 4.5, "cnt" : 27301 }
{ "_id" : 4, "cnt" : 35969 }
{ "_id" : 3.5, "cnt" : 35008 }
{ "_id" : 3, "cnt" : 25996 }
{ "_id" : 2.5, "cnt" : 18843 }
{ "_id" : 2, "cnt" : 11426 }
{ "_id" : 1.5, "cnt" : 4976 }
{ "_id" : 1, "cnt" : 4874 }

```

11.

```

db.review.aggregate([{$limit: 500000}, {$out: "Subreview"}])

```

```
> db.review.aggregate([{$limit: 500000}, {$out: "Subreview"}])
> show collections
Subreview
business
review
test_map_reduce
user
>
> db.Subreview.count()
500000
```

db.Subreview.createIndex({'text':'text'})

```
> db.Subreview.createIndex({'text':'text'})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```

db.Subreview.createIndex({'useful':1})

```
> db.Subreview.createIndex({'useful':1})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 2,
  "numIndexesAfter" : 3,
  "ok" : 1
}
```

db.Subreview.find({'useful':{'\$gt':9}, '\$text':{'\$search':'delicious'}})

```
> db.Subreview.find({'useful':{'$gt':9}, '$text':{'$search':'delicious'}})
{ "_id" : "600d7ea5f5eb9d17c3a80d1", "review_id" : "ECLxSkdKPF572zBF9H21060", "user_id" : "R_xjzm2nZgyOkqFKPkKjNA", "business_id" : "RBACl6hgGyoaTRv0KkUjPA", "stars" : 5, "useful" : 19, "funny" : 16, "cool" : 18, "text" : "Hawaii in Las Vegas, Aloha!\n\nOne of my yelp friend (Sandra S) raved about this place on her last visit to sin city. I love Hawaiian food so I bookmarked it knowing I was going in December. Fast forward a few months and I'm in Hawaiian Heaven. \n\nHere's the run down:\nFried Mac n Cheese Balls - a good starter. I tried it was fried perfectly and the mac n cheese was so creamy inside. It was cheap too. $2.45 for 4 big pieces. Delicious \n\nWe came here for the Poke, we got:\nShoyu Ahi poke - my favorite. Delicious. \nAhi Furukake poke - it was still a bit frozen, but still good. \nSpicy Ahi Combo w/ Crabs, Avocados & Cucumber - my favorite 1A. Delicious\nTako poke w/ Limu (seaweed) - octopus was super tender. \n\nThe Ahi's were big chunks of raw tuna. So fresh, so good. You've got to try this place if you like poke. \n\nWe also got some hot dishes.\nLoco Moco - we were all loco for this dish. The best loco moco I've ever had. The hamburger was just so flavorful and you get two big hand made patties. The gravy is just money. With two eggs & two scoops of rice. Oh man, this alone made the trip here so worth it. Delicious. Must order here. \n\nKalu Pig & Cabbage - another winner. Authentic Hawaiian dish. No fake smoke, tender, juicy goodness. Yum yum yum. \n\nKali - Delicious. fall off the bone tender and the marinade was spot on. Cooked to perfection. Delicious. \n\nHilo Bay Bento Special - this had everything. A piece of spam, pork cutlet, pork teriyaki, beef short rib, noodles, pickled radish and all on a bed of rice. This dish is a mouthful and everything was so so good. It's a special, so they might not have it everyday, but if they do, get it. You won't regret it. \n\nThe service was very friendly and helpful. We ordered a lot of food and kept adding to the order and the guy taking the order was very patient with us. \n\nThe place is small. A few tables (for 2 only) line the wall. Get your food to go. \n\nI need a nap now after th at luau we just had. \n\nHawaiian food done right. \n\nSandra, thank you for the recommendation! \n\nI'll definitely be back.", "date" : "2013-12-10 01:05:30" }
{ "_id" : "600d7ea7f5eb9d17c3b10f1", "review_id" : "0Gwrt9S0u0U7z8X0MSH1", "user_id" : "Pf22rldottzr1Zz8B9v1Q", "business_id" : "d4ufg0Lh_VK8tLU8R5vyp", "stars" : 5, "useful" : 22, "funny" : 15, "cool" : 21, "text" : "I Only Have Pies for You!\n\nParody of 'Superstition', Stevie Wonder)\n\nTotally delicious, side of mac n' cheese\n\nPiled with chili potle pesto, add some bacon, please\n\nMinestrumbo soup yeah, is it a soup or stew?\n\nMeatballs, zuchs, and sausage, in a tomato brew\n\nTotally delicious, pasta, \nCoop de Ville\n\nChix on fettuccini, fills me to the gills\n\nLady is a Scampi", "date" : "2014-05-12 02:25:46" }
{ "_id" : "600d7ea7f5eb9d17c42d92", "review_id" : "9ePxnioVlgYKf-5RvMSag", "user_id" : "Gjz_frs_AP2wMbj9g0pA", "business_id" : "CHdJLXlKwMx8KpDwE6B_A", "stars" : 4, "useful" : 11, "funny" : 0, "cool" : 5, "text" : "My wife and I had been to the other Chino Bandido a few years ago. I saw that this place was within reasonable driving distance from w ork (4 miles) and since we have hour lunches, I decided to drive down. I walked in, filled out my card on what I wanted (for those that don't know...Chino Bandido is a fusion combin e that allows you to mix and match different items. You fill out a card with a pen and write down what you want...endless combinations are available pretty much) and then paid the m e at my money. A few minutes later, I had my sack of food and was on my way out the door. Here is what I ordered.....\n\nI went with a 2 item combo and decided to get 2 q uesadillas (Jen Red Pork and Jale Red Chicken) Black beans and pork fried rice. The total for this huge monstrosity was $9.10. Here is how everything tasted.....\n\nJen Red Pork Quesadilla - This thing was delicious. It was pretty much sweet and sour pork wrapped in a tortilla with cheese. They kind of 'fry/riddle' the tortillas so they have a bit of a crunch/crisp to them. They stuffed this to the gills with tons of crunchy/sweet/hot pork.\n\nJade Red Chicken - Very similar to the Jen Red Pork except that it was chicken. Th ink sweet and sour chicken. I was very impressed with my 2 quesadilla combo. It was almost too much though. I wound up stuffed and couldn't even finish two 1/4s of each quesadilla. \n\nBlack Beans - I'm usually not a fan of beans. These beans were absolutely delicious though. They are as black as the night sky and they came covered with a bunch of cheese. The y reminded me of a sort of refried bean but way more delicious. It was almost like a giant bean dip. Alternating bites of beans and quesadilla was a killer combo.\n\nPork Fried Rice - This was pretty good. I left a lot of it in the container just because I was so damn full from everything else. This was a moister fried rice with little pieces of pork in it. W as it the best fried rice ever? Nah. Did it work with the combo of everything else? Yeah. \n\nSnickerdoodle - They hook you up with a cookie for dessert. The super cookie was del icious. Soft and chewy and sweet. It was the first cookie I have had in months.\n\nEverything was really tasty! You could probably split a 2 item combo between 2 people. I tried m y darndest to finish everything and didn't even come close. They definitely hook up the portions! I will definitely be back to try out more combos. The choices are endless and t he food is delicious. Service was friendly too. I'm glad this place is nearby.", "date" : "2014-05-12 02:25:46" }
{ "_id" : "600d7ea7f5eb9d17c40d649", "review_id" : "0pX8T08g0n3v0GCM5w", "user_id" : "MMf0hE5K5Gd1LVN7zCdnA", "business_id" : "8ZqH2jwtnca3NAwFwMTv20", "stars" : 4, "useful" : 14, "funny" : 4, "cool" : 8, "text" : "Breakfast for Lunch\n\nThis was the first time I visited 'Scramble a Breakfast Joint' based on the reviews of 'Yelpers'. Fro m the moment you walk in the door, you are greeted by a friendly staff member and you can't help but notice the cleanliness of the restaurant. \n\nThe menu is printed on the wall, so you place your order at the register, take a seat, and wait for your delicious meal to be delivered. There is a buttonless coffee and beverage station in the middle of the restaurant and TVs throughout the entire restaurant.\n\nOn this visit I had the buttermilk pancakes and bacon all I have to say is, these pancakes were not only delicious but made perfectly, sof t and fluffy on the inside and golden brown on the outside, perfect! and the bacon was thick, but not too thick; crispy but not too crispy; once again, perfect, tasty and delicious. \n\nI look forward to revisiting 'Scramble a Breakfast Joint' again and try other delicious items offered on their menu.", "date" : "2017-07-28 12:45:52" }
{ "_id" : "600d7ea7f5eb9d17c3a80d1", "review_id" : "tLNMWVj84dd8K9jPb", "user_id" : "3nUSC2k5T_2mWMLN7bW", "business_id" : "VTLWgPhyIvIC_LNVf0bw", "stars" : 5, "useful" : 12, "funny" : 3, "cool" : 10, "text" : "Over the years I've wished that GVR would get more restaurants since the location is so convenient and finally it's happening. Bottl g ia isn't just a restaurant, it's a great restaurant. \n\nReservations seem to be a must. Our first attempt to come here was without a reservation and they were booked for the entire e vening. This time I made a reservation on Open Table for a 6:00 dinner. The restaurant was pretty full this time as well. \n\nThe restaurant is bright and cheery, it's much more open t han the last Italian restaurant that was in this spot. The restaurant is beautifully decorating and very inviting. Next to the hostess stand there are couches for lounge seating, a la rge bar, tables, booths, and two outdoor patios with seating. In the back there is a private room that can seat a large group. \n\nWe were brought the bread board to start. The NURMS
```

```
> db.Subreview.find({'useful':{'$gt':9}, '$text':{'$search':'delicious'}}).count()
911
```


12.

```
db.Subreview.aggregate([
{$match: {'useful':{$gt:6}, 'funny':{$gt:6}, 'cool':{$gt:6}}},
{$group: {'_id': '$business_id', 'stars_avg': {$avg: '$stars'}}},
{$sort: {'_id': -1}}
])
```

```
> db.Subreview.aggregate([ {$match: {'useful':{$gt:6}, 'funny':{$gt:6}, 'cool':{$gt:6}}}, {
$gdb.Subreview.aggregate([ {$match: {'useful':{$gt:6}, 'funny':{$gt:6}, 'cool':{$gt:6}}}, {
$group: {'_id': '$business_id', 'stars_avg': {$avg: '$stars'}}}, {$sort: {'_id': -1}} ]])
{ "_id" : "zvQIEpJUMLLmMMffNntHXQ", "stars_avg" : 2 }
{ "_id" : "zrYpLdnGKA_EmOhgRCy_vg", "stars_avg" : 4.5 }
{ "_id" : "znWHLW1pt19HzW1VY6KfCA", "stars_avg" : 4 }
{ "_id" : "znBAFVFRMvdUvJ7eJY_vjA", "stars_avg" : 3 }
{ "_id" : "zitXLajbET0uHQJfok4a4g", "stars_avg" : 4 }
{ "_id" : "zidkKI_N10PxiddT0QH_Q", "stars_avg" : 4 }
{ "_id" : "zenxm7shqn60Cfw_2PGbrg", "stars_avg" : 4 }
{ "_id" : "zcpf0E1tBET_5YUjq8y1lg", "stars_avg" : 4 }
{ "_id" : "zauhMY78k36XPfxD3GURkQ", "stars_avg" : 3.5 }
{ "_id" : "zYRBRfYuq_6x-wNka8NqrA", "stars_avg" : 4 }
{ "_id" : "zYKblyjRDoi0um6NYuvZ6g", "stars_avg" : 1 }
{ "_id" : "zWB6hcE7Pxwv0VoI2fYIWw", "stars_avg" : 4 }
{ "_id" : "zSc_PmocVDJEtQwin0ts2w", "stars_avg" : 3 }
{ "_id" : "zPxwTy_2W0slWtKCr8B1nQ", "stars_avg" : 1 }
{ "_id" : "zKqWjDZHo-lk7_E3hHkORg", "stars_avg" : 4 }
{ "_id" : "zHJhV-P_FesPreukKMo0eA", "stars_avg" : 1.5 }
{ "_id" : "zEq6RxjGkyf9AzUFtBvc0g", "stars_avg" : 4 }
{ "_id" : "zBtR7328Vuts_7B9qm3DVQ", "stars_avg" : 5 }
{ "_id" : "z9oJeVmNEc3F0ToZ0x4WuQ", "stars_avg" : 4.666666666666667 }
{ "_id" : "z8Em-bhZI3Mmspl7tj6tg", "stars_avg" : 2 }
Type "it" for more
```

统计结果数量:

```
db.Subreview.aggregate([
{$match: {'useful':{$gt:6}, 'funny':{$gt:6}, 'cool':{$gt:6}}},
{$group: {'_id': '$business_id', 'stars_avg': {$avg: '$stars'}}},
{$sort: {'_id': -1}},

{$count: "TotalCount"}

])
```

```
> db.Subreview.aggregate([ {$match: {'useful':{$gt:6}, 'funny':{$gt:6}, 'cool':{$gt:6}}}, {
$group: {'_id': '$business_id', 'stars_avg': {$avg: '$stars'}}}, {$sort: {'_id': -1}}, {$count: "Total
Count"} ]])
{ "TotalCount" : 1558 }
```

13.

```
db.business.createIndex({'loc': "2dsphere"})
```

```
> db.business.createIndex({'loc': "2dsphere"})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 2,
  "numIndexesAfter" : 3,
  "ok" : 1
}
```

查询目标商家的坐标:

```
> db.business.find({'business_id':'xvX2CttrVhyG2z1dFg_0xw'})
{ "_id" : ObjectId("6016c6b4af81085b0f2183c4"), "business_id" : "xvX2CttrVhyG2z1dFg_0xw",
"name" : "Farmers Insurance - Paul Lorenz", "address" : "15655 W Roosevelt St, Ste 237", "
city" : "Goodyear", "state" : "AZ", "postal_code" : "85338", "latitude" : 33.4556129678, "
longitude" : -112.3955963552, "stars" : 5, "review_count" : 3, "is_open" : 1, "attributes"
: null, "categories" : [ "Insurance", "Financial Services" ], "hours" : { "Monday" : "8:0
-17:0", "Tuesday" : "8:0-17:0", "Wednesday" : "8:0-17:0", "Thursday" : "8:0-17:0", "Friday
" : "8:0-17:0" }, "loc" : { "type" : "Point", "coordinates" : [ -112.3955963552, 33.455612
9678 ] } }
```

```
db.business.find(
{loc: {$near:{$geometry:{type:"Point", coordinates:[-112.3955963552, 33.4556129678]}},
$maxDistance: 100}}},
{'name':1, 'address':1, 'stars':1, '_id':0}
)
```

```
> db.business.find( {loc: {$near:{$geometry:{type:"Point", coordinates:[-112.3955963552, 3
3.4556129678]}}, $maxDistance: 100}}}, {'name':1, 'address':1, 'stars':1, '_id':0} )
{ "name" : "Farmers Insurance - Paul Lorenz", "address" : "15655 W Roosevelt St, Ste 237",
"stars" : 5 }
{ "name" : "Kurt Bojarski - American Family Insurance", "address" : "15655 W Roosevelt St
, Ste 101", "stars" : 5 }
```

14.

首先创建一个关于用户id和评论日期的索引

```
db.Subreview.createIndex({user_id: 1, date: 1})
```

然后聚合查询

```
db.Subreview.aggregate([
{$match: {'date': {$gte: "2017-01-01"}}},
{$group: {_id: "$user_id", total: {$sum: 1}}},
{$sort: {total: -1}},
{$limit: 20}
])
```

```
> db.Subreview.aggregate([ {$match: {'date': {$gte: "2017-01-01"}}}, {$group: {_id: "$user_id"
, total: {$sum: 1}}}, {$sort: {total: -1}}, {$limit: 20} ])
{ "_id" : "I-4KVZ9lqHhk8469X9FvhA", "total" : 61 }
{ "_id" : "qKpkRCPk4ycb1lTfFcRbNw", "total" : 46 }
{ "_id" : "bLbSNkLggFnqWNNzzq-Ijw", "total" : 42 }
{ "_id" : "ic-tyiljELL_umxZVh8KNA", "total" : 37 }
{ "_id" : "tFSnxUGaNa2pwb4zClBu_Q", "total" : 32 }
{ "_id" : "z9w399cBpCAKXhH_JA1AtQ", "total" : 32 }
{ "_id" : "nyl_1VcRIAYI55bb_scpdw", "total" : 31 }
{ "_id" : "zPByj_3y6TBok5BpCyYrTw", "total" : 31 }
{ "_id" : "YE54kKTuqJJPNYWIkiP0EQ", "total" : 30 }
{ "_id" : "xDl9ZF3SckkZde_48W6WeA", "total" : 29 }
{ "_id" : "PcvbB0C0cs6_suRDH7TSTg", "total" : 29 }
{ "_id" : "f_5VRh79aew1cVwUmC1PJA", "total" : 29 }
{ "_id" : "6Ki3bAL0wx9ymbdJqbSWMA", "total" : 28 }
{ "_id" : "PGeiszoVusiv0wTHVdWkLA", "total" : 28 }
{ "_id" : "keBv05MsMFBd0Hu98vXThQ", "total" : 27 }
{ "_id" : "QbDJB-4XAxn3BGwi6Rrzjw", "total" : 26 }
{ "_id" : "wNLZnNNLV8r0GiPjqMPVdQ", "total" : 26 }
{ "_id" : "3nIuSCZk5f_2wWYMLN7h3w", "total" : 25 }
{ "_id" : "ELcQDlf69kb-ihJfxZyL0A", "total" : 25 }
{ "_id" : "U95wccXN_J8JwA5Ktlu8tw", "total" : 25 }
```

15.

```
db.business.mapReduce(
```



```

function(){
emit(this.business_id, {stars: this.stars, count:1});
},

function(key, values){
var totalStars=0;
var totalCount=0;
values.forEach(function(value) {
totalStars+=value.stars;
totalCount+=value.count;
});
return {stars:totalStars, count:totalCount};
},
{
out:"test_map_reduce",
finalize: function(key, reducedValue){
var avgStars = reducedValue.stars / reducedValue.count;
return { count: reducedValue.count, stars: reducedValue.stars, avg: avgStars };
}
}
)

```

```
{ "result" : "test_map_reduce", "ok" : 1 }
```

```

> show collections
Average_Stars
Subreview
business
review
test_map_reduce
user

```

```

> db.test_map_reduce.find()
{ "_id" : "6zKU06XecBdRDnzgNB5NmQ", "value" : { "count" : 1, "stars" : 2.5, "avg" : 2.5 } }
{ "_id" : "a8RfC0r41Jgp7urWbKSAzg", "value" : { "count" : 1, "stars" : 2.5, "avg" : 2.5 } }
{ "_id" : "ZpPd9a-Tv_NGXh8P1ivjzA", "value" : { "count" : 1, "stars" : 2.5, "avg" : 2.5 } }
{ "_id" : "Jvx8CyM8om6tZvn8PX0WEA", "value" : { "count" : 1, "stars" : 3.5, "avg" : 3.5 } }
{ "_id" : "07uzVmRB8GDMU8XaX5l7Eg", "value" : { "count" : 1, "stars" : 3, "avg" : 3 } }
{ "_id" : "1-QYao7fyZ0ihnM16DN9hw", "value" : { "count" : 1, "stars" : 4.5, "avg" : 4.5 } }
{ "_id" : "6a6IaS9T4uLUR6s7ak3hJQ", "value" : { "count" : 1, "stars" : 5, "avg" : 5 } }
{ "_id" : "kp9o9uv_scnT6ZBdDvvLaA", "value" : { "count" : 1, "stars" : 4, "avg" : 4 } }
{ "_id" : "BCORM4dE1eJXnyPXoaH3HQ", "value" : { "count" : 1, "stars" : 4, "avg" : 4 } }
{ "_id" : "u09bcA8mmWEAusKBudAidg", "value" : { "count" : 1, "stars" : 4.5, "avg" : 4.5 } }
{ "_id" : "w_0nQYHhXhrSR95pWZafVg", "value" : { "count" : 1, "stars" : 4, "avg" : 4 } }
{ "_id" : "r6tNDARvVhHcRD88DEND0Q", "value" : { "count" : 1, "stars" : 4, "avg" : 4 } }
{ "_id" : "mVDXeHkpWK-GNqdi7fN4MQ", "value" : { "count" : 1, "stars" : 3, "avg" : 3 } }
{ "_id" : "mA_XSJGBwGm9RgHndazwiw", "value" : { "count" : 1, "stars" : 3, "avg" : 3 } }
{ "_id" : "xi4yGWRFBRaLDygBj2lqBw", "value" : { "count" : 1, "stars" : 2.5, "avg" : 2.5 } }
{ "_id" : "laac2uH1lQVzBjKFUjuA1Q", "value" : { "count" : 1, "stars" : 3, "avg" : 3 } }
{ "_id" : "0jAexNHRHhkQv4zNYqenBA", "value" : { "count" : 1, "stars" : 3, "avg" : 3 } }
{ "_id" : "MAKV_ZbhTcVA-IcdjKYhRQ", "value" : { "count" : 1, "stars" : 2.5, "avg" : 2.5 } }
{ "_id" : "RhGZ1T3d8hodfvuZER7pIg", "value" : { "count" : 1, "stars" : 5, "avg" : 5 } }
{ "_id" : "kQIFaTQW1Q4Lc8ksvyT9QQ", "value" : { "count" : 1, "stars" : 4, "avg" : 4 } }
Type "it" for more

```

Lab 3





1.

MATCH (city:CityNode)

RETURN city

LIMIT 10

```
yelp$ MATCH (city:CityNode) RETURN city LIMIT 10
```

 Graph	"city"
 Table	{"city": "Moon Township"}
 Text	{"city": "montreal"}
 Code	{"city": "Uxbridge"}
	{"city": "Warrensville Hts."}
	{"city": "Warrensville"}
	{"city": "Phoniex"}
	{"city": "Norton"}
	{"city": "Ahwatukee"}
	{"city": "Eastlake"}
	{"city": "Huntingdon"}

2.

```
MATCH (business:BusinessNode {city:'Ambridge'})  
RETURN business
```

```
yelp$ MATCH (business:BusinessNode {city:'Ambridge'}) RETURN business
```

Graph	"business"
Table	
Text	{ "reviewcount": "3", "address": "400 Merchant St", "city": "Ambridge", "latitude": "40.5840997", "businessid": "dJ0R-XT78LUQeNHQkD-G9g", "name": "Ice Cream Therapy", "stars": "3.5", "longitude": "-80.225135" }
Code	{ "reviewcount": "4", "address": "304 Duss Ave", "city": "Ambridge", "latitude": "40.5823797", "businessid": "3gL18eXylqutlzqb6TmB0w", "name": "Action Tire Company", "stars": "5.0", "longitude": "-80.2238749" }
	{ "reviewcount": "18", "address": "603 Duss Ave", "city": "Ambridge", "latitude": "40.5879393", "businessid": "Q_0eGl-aElqHKukHvmLdwA", "name": "Nelia's Smokehouse", "stars": "4.0", "longitude": "-80.2248851" }
	{ "reviewcount": "14", "address": "755 Merchant St", "city": "Ambridge", "latitude": "40.5886057", "businessid": "Eu_zPTrNVAXkpdSxf7CJ2w", "name": "K & N Restaurant", "stars": "4.5", "longitude": "-80.2291033" }
	{ "reviewcount": "4", "address": "663 Merchant St", "city": "Ambridge", "latitude": "40.5876498", "businessid": "Yjf0i2J9q52dYIT8UVGT3g", "name": "Heritage Floral Shoppe", "stars": "4.5", "longitude": "-80.2284934" }
	{ "reviewcount": "5", "address": "598 Merchant St", "city": "Ambridge", "latitude": "40.5865326", "businessid": "y3IVqEFHmrkgVKj2x1Ci4w", "name": "Off the Hook Exotics", "stars": "4.0", "longitude": "-80.2269877" }
	{ "reviewcount": "32", "address": "1007 Merchant St", "city": "Ambridge", "latitude": "40.592077", "businessid": "729grSa1Wsn-hfv7D5uOxg", "name": "Pizza House", "stars": "4.5", "longitude": "-80.230377" }

3.

```
MATCH (r:ReviewNode {reviewid:'rEITo90tpyKmEfNDp3Ou3A'})-[:Reviewed]->
(business:BusinessNode)
RETURN business
```

```
yelp$ MATCH (r:ReviewNode {reviewid:'rEITo90tpyKmEfNDp3Ou3A'})-[:Reviewed]->(business:BusinessNode) RETURN business
```

Graph	"business"
Table	
Text	{ "reviewcount": "106", "address": "2800 E Germann Rd", "city": "Chandler", "latitude": "33.278104", "businessid": "6Lj2BJ4tJeu7db5asGHQ4w", "name": "Slim Chickens", "stars": "3.0", "longitude": "-111.7924641" }
Code	

4.

```
MATCH (user:UserNode)-[:Review]->(:ReviewNode)-[:Reviewed]->(:BusinessNode
{businessid:'fyJAqmweGm8VXnpU4CWGNw'})
RETURN user.name, user.fans
```

```
yelp$ MATCH (user:UserNode)-[:Review]→(:Rev
```



Table

A

Text



Code

"user.name"	"user.fans"
"Joseph"	"0"
"Linda"	"0"
"Brittanie"	"24"
"Melinda"	"17"
"Kellie"	"1"
"Calinda"	"0"
"Xandon"	"0"
"Teresa"	"78"
"Island"	"1"
"John"	"2"
"Chris"	"0"
"Hicks"	"0"
"Lucy"	"0"
"Shawna"	"1"
"Rochelle"	"0"

```
MATCH (:UserNode {userid:'TetzbpqA2BFBrC0y0sCbfw'})-[:Review]->(:ReviewNode {stars:'5.0'})-[:Reviewed]->(business:BusinessNode)
RETURN business.name, business.address
```

```
yelp$ MATCH (:UserNode {userid:'TetzbpqA2BFBrC0y0sCbfw'})-[:Review]->(:ReviewNode {stars:'5.0'})-[:Reviewed]->(business:BusinessNode) RETURN business.name, business.address
```

	business.name	business.address
1	"The Buffet"	"3131 Las Vegas Blvd S"
2	"MGM Grand Hotel"	"3799 Las Vegas Blvd S"
3	"Paris Las Vegas Hotel & Casino"	"3655 Las Vegas Blvd S"
4	"Blue Man Group"	"3900 Las Vegas Blvd S"
5	"Burger Bar"	"3930 S Las Vegas Blvd, Ste 121A"
6	"Bellagio Hotel"	"3600 S Las Vegas Blvd"
7	"The Venetian Las Vegas"	"3355 South Las Vegas Boulevard"

6.

```
MATCH (business:BusinessNode)
RETURN business.name, business.stars, business.address
ORDER BY business.stars DESC
LIMIT 15
```

```
yelp$ MATCH (business:BusinessNode) RETURN business.name, business.stars, business.address ORDER BY business.stars DESC LIMIT 15
```

	business.name	business.stars	business.address
1	"Wesley Friedman - FBC Mortgage"	"5.0"	"6775 Edmond St, Ste 210"
2	"Stephen L Walker, DDS, MS"	"5.0"	"2220 W Southern Ave, Ste 102, Endodontic Specialists Ltd"
3	"Knot Salon"	"5.0"	"4848 E Cactus Rd, Ste 100"
4	"Brian's Furniture"	"5.0"	"30808 Center Ridge Rd"
5	"Three Square"	"5.0"	"4190 N Pecos Rd"
6	"Key To Healing Massage"	"5.0"	"14202 N Scottsdale Rd"
7	"Farmers Insurance - Paul Lorenz"	"5.0"	"15655 W Roosevelt St, Ste 237"
8	"Vita Bella Fine Day Spa"	"5.0"	"5940 W Union Hills Dr"
9	"Maurices"	"5.0"	"7981 W Tropical Pkwy, Ste 120"
10	"OzBraz"	"5.0"	"560 E Germann Rd, Ste 107"
11	"Myron Hensel Photography"	"5.0"	"

Started streaming 15 records after 1 ms and completed after 161 ms.

7.

```
MATCH (user:UserNode)
WHERE tointeger(user.fans)>200
RETURN user.name, user.fans
LIMIT 10
```

```
yelp$ MATCH (user:UserNode) WHERE tointeger(user.fans)>200 RETURN user.name, user.fans LIMIT 10
```

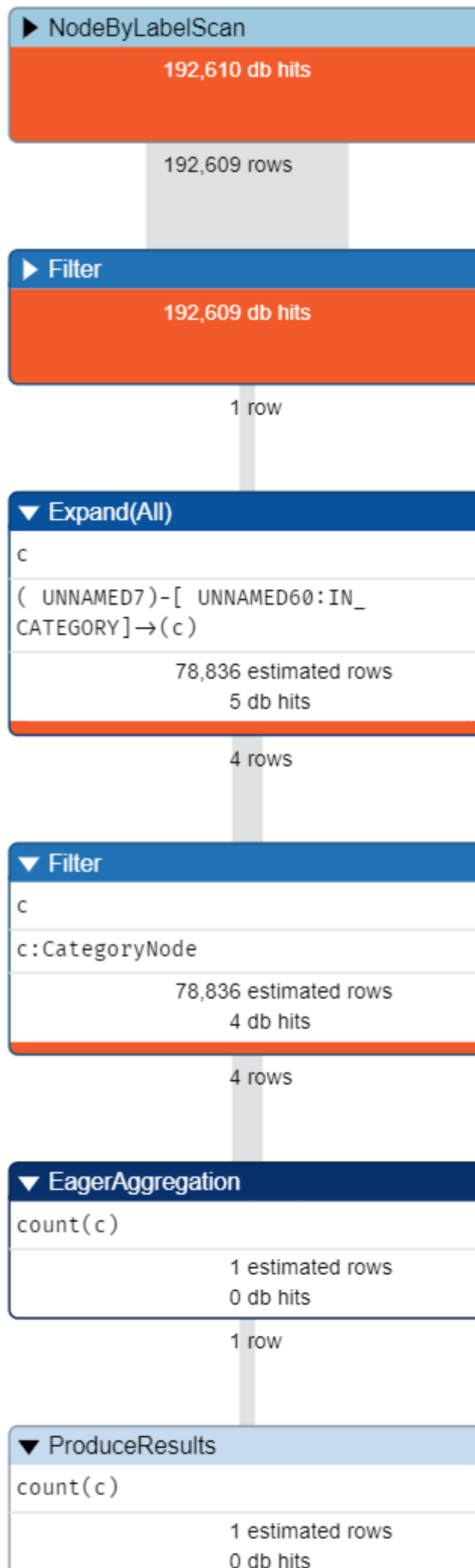
	user.name	user.fans
1	"Keane"	"696"
2	"Annie"	"407"
3	"Jason"	"251"
4	"Regina"	"231"
5	"Alma"	"272"
6	"Stephanie"	"282"
7	"Michael"	"912"
8	"Nette"	"426"
9	"Alan"	"379"
10	"Keila"	"447"

8.

```
MATCH (:BusinessNode {businessid:'tyjquHslrAuF5EUejbPfrw'})-[:IN_CATEGORY]->
(c:CategoryNode)
RETURN count(c)
```

使用PROFILE查看执行计划:

```
PROFILE MATCH (:BusinessNode {businessid:'tyjquHslrAuF5EUejbPfrw'})-[:IN_CATEGORY]->
(c:CategoryNode)
RETURN count(c)
```



9.

1 row

MATCH (business:BusinessNode {businessid:'tyjquHslrAuF5EUejbPfrw'})-[:IN_CATEGORY]->(c:CategoryNode)
RETURN collect(c.category) as category

Result

category

1

[" Latin American", " Venezuelan", " Restaurants", "Mexican"]

10.

MATCH(:UserNode {name:'Allison'})-[:HasFriend]->(friend:UserNode)
WITH friend.name as friendsList, size((friend) -[:HasFriend]->()) as numofFOFs
RETURN friendsList, numofFOFs

	friendsList	numofFOFs
1	"David"	6
2	"Regina"	32
3	"Michael"	292
4	"Greg"	73
5	"Jennifer"	375
6	"Jerry"	85
7	"Michael"	450

Started streaming 19478 records in less than 1 ms and completed after 53 ms, displaying first 1000 rows.

11.

MATCH (business:BusinessNode)-[:IN_CATEGORY]->(:CategoryNode {category:'Salad'})
WITH business.city as city, count(*) as count
RETURN city, count
ORDER BY count DESC
LIMIT 5

	city	count
1	"Toronto"	52
2	"Charlotte"	40
3	"Phoenix"	39
4	"Las Vegas"	39
5	"Pittsburgh"	19

12.

```

MATCH (business:BusinessNode)
WITH business.name as name, count(*) as cnt
RETURN name, cnt
ORDER BY cnt DESC
LIMIT 10

```

	name	cnt
1	"Starbucks"	1066
2	"McDonald's"	806
3	"Subway"	768
4	"Tim Hortons"	333
5	"Pizza Hut"	320
6	"Taco Bell"	313
7	"Burger King"	302
8	"Walgreens"	301
9	"Wendy's"	294
10	"The UPS Store"	279

13.

```

MATCH (businessall:BusinessNode)
WITH count(distinct businessall.name) as cnt
MATCH (business:BusinessNode)
WHERE tointeger(business.reviewcount) > 5000
WITH cnt, count(business.name) as subcnt, business.name as name, business.reviewcount as reviewcount
RETURN subcnt*1.0/cnt, name, reviewcount
ORDER BY reviewcount DESC

```

	subcnt*1.0/cnt	name	reviewcount
1	0.000006894364546419757	"Mon Ami Gabi"	"8348"
2	0.000006894364546419757	"Bacchanal Buffet"	"8339"
3	0.000006894364546419757	"Wicked Spoon"	"6708"
4	0.000006894364546419757	"Hash House A Go Go"	"5763"
5	0.000006894364546419757	"Gordon Ramsay BurGR"	"5484"
6	0.000006894364546419757	"Earl of Sandwich"	"5075"

14.

```

MATCH (business:BusinessNode)-[:IN_CATEGORY]->(:CategoryNode {category:'Zoos'})
WITH business
MATCH (:ReviewNode {stars:'5.0'})-[:Reviewed]->(business:BusinessNode)
RETURN distinct business.city

```

business.city

1	"Toronto"
2	"Cave Creek"
3	"Las Vegas"
4	"Calgary"

15.

```

MATCH (user:UserNode)-[:Review]->(:ReviewNode)-[:Reviewed]->(business:BusinessNode)
WITH count(distinct user) as user_count, business
RETURN business.businessid, business.name, user_count
ORDER BY user_count DESC
LIMIT 10

```

	business.businessid	business.name	user_count
1	"4JNXUY8wbaaDmk3BPzIWw"	"Mon Ami Gabi"	8349
2	"RESDUcs7flihp38-d6_6g"	"Bacchanal Buffet"	8341
3	"K7IWdNUhCbcnEvl0NhGewg"	"Wicked Spoon"	6710
4	"f4x1YBxkLrZg652xt2KR5g"	"Hash House A Go Go"	5763
5	"cYwJA2A6l12KNkm2rtXd5g"	"Gordon Ramsay BurGR"	5484
6	"DkYS3arLOhA8si5uUEmHOW"	"Earl of Sandwich"	5076
7	"2weQS-RnoOBhb1KsHKycSQ"	"The Buffet"	4401
8	"5LNZ67Yw9RD6nf4_UhXOjw"	"The Cosmopolitan of Las Vegas"	4322
9	"ICQpiavjPzJ5_3gPD5Ebg"	"Secret Pizza"	4286
10	"ujHlaprwCQ5ewziu0Vi9rw"	"The Buffet at Bellagio"	4230

16.

准备工作，使用语句为用户Node新建一个属性flag用来实验

MATCH (user:UserNode)

SET user.flag = user.fans

ERROR ServiceUnavailable

Could not perform discovery. No routing servers available. Known routing table: RoutingTable[database=yelp, expirationTime=1698241200886, currentTime=1698241027220, routers=[], readers=[], writers=[]]

```
OpenJDK 64-Bit Server VM warning: INFO: os::commit_memory(0x00000000ffa00000, 1048576, 0) failed; error='Not enough space' (errno=12)
#
# There is insufficient memory for the Java Runtime Environment to continue.
# Native memory allocation (mmap) failed to map 1048576 bytes for committing reserved memory.
# An error report file with more information is saved as:
# /root/neo4j-community-4.0.9/bin/hs_err_pid1956.log
```

由于服务器内存不够而报错。因此需要对属性作用的范围做出一定的限定，例如fans大于400，于是属性创建完成

MATCH (user:UserNode)

WHERE toInteger(user.fans) > 400

SET user.flag = user.fans

Set 295 properties, completed after 1617 ms.

对用户Node的flag属性执行查询（flag>8000）、创建（flag=10000）、更新（flag=456 to flag=3456）、删除（flag>4000）操作

查询：

MATCH (user:UserNode)

WHERE toInteger(user.flag) > 8000

RETURN user

"user"
{ "flag": "9538", "cool": "13227", "name": "Mike", "userid": "37cpUoM8h1kSQfReIEBd-Q", "useful": "19715", "funny": "10085", "fans": "9538" }

Started streaming 1 records after 2 ms and completed after 713 ms.

创建:

```
MATCH (user:UserNode)
WHERE toInteger(user.fans) = 198
SET user.flag = 10000
```

Set 8 properties, completed after 643 ms.

更新:

```
MATCH (user:UserNode)
WHERE toInteger(user.flag) = 456
SET user.flag = 3456
```

Completed after 495 ms.

删除:

```
MATCH (user:UserNode)
WHERE user.flag > 4000
REMOVE user.flag
```

Set 8 properties, completed after 12 ms.

重置属性信息（最上面的步骤），然后为UserNode的属性flag建立索引

```
CREATE INDEX FOR (user:UserNode) ON (user.flag)
```

Added 1 index, completed after 15 ms.

对UserNode的flag属性执行相同的查询、创建、更新、删除操作，比较操作时间

查询:

Started streaming 1 records in less than 1 ms and completed after 507 ms.

创建:

Set 8 properties, completed after 731 ms.

更新:

Set 1 property, completed after 584 ms.

删除:

Set 11 properties, completed after 673 ms.

17.

```
MATCH (user1:UserNode {userid: 'tvZKPah2u9G9dFBg5GT0eg'})-[:Review]->(:ReviewNode)-
[:Reviewed]->(b:BusinessNode)
WITH user1, COLLECT(DISTINCT b) AS user1_businesses
MATCH (user2:UserNode)-[:Review]->(:ReviewNode)-[:Reviewed]->(b:BusinessNode)
WHERE NOT EXISTS ((user1)-[:HasFriend]->(user2)) AND b IN user1_businesses AND user1 <> user2
WITH user1, user2, COUNT(DISTINCT b) AS num
RETURN user1.name, user2.name, num
ORDER BY num DESC
```

	user1.name	user2.name	num
1	"Emily"	"Anthony"	10
2	"Emily"	"Norm"	10
3	"Emily"	"Michael"	8
4	"Emily"	"Rodney"	8
5	"Emily"	"Janice"	8
6	"Emily"	"Lorrie"	7
7	"Emily"	"Darrell"	7
8	"Emily"	"J"	7

▼ NodeIndexSeek
user1
:UserNode(userid)
Ordered by user1.userid ASC
1 estimated rows
2 db hits

1 row

▼ Expand(All)
user1
(user1)-[UNNAMED58:Review]->(UNNAMED70)
Ordered by user1.userid ASC
4 estimated rows
16 db hits

15 rows

▼ Filter
user1
`anon[70]`:ReviewNode
Ordered by user1.userid ASC
4 estimated rows
15 db hits

15 rows

▼ Expand(All)
user1
(UNNAMED70)-[UNNAMED83:Reviewed]->(b@97)
Ordered by user1.userid ASC
4 estimated rows
30 db hits

15 rows

▼ Filter
user1
`b`:BusinessNode
Ordered by user1.userid ASC
4 estimated rows
15 db hits

15 rows

▼ NodeByLabelScan
user1, user1_businesses, user2
:UserNode
3,308,435 estimated rows
1,637,139 db hits

1,637,138 rows

▼ Argument
user1, user2
2,977,591 estimated rows
0 db hits

1,637,137 rows

▼ Filter
user1, user1_businesses, user2
not user1 = user2
2,977,591 estimated rows
0 db hits

1,637,137 rows

▼ Expand(Into)
user1, user2
(user1)-[REL269:HasFriend]->(user2)
16 estimated rows
16,390,875 db hits

0 rows

▼ EagerAggregation
user1, user1_businesses
2 estimated rows
0 db hits

1 row

▼ AntiSemiApply
user1, user1_businesses, user2
744,398 estimated rows
0 db hits

1,637,032 rows

▼ Apply
user1, user1_businesses, user2
2,280,032 estimated rows
0 db hits

1,637,032 rows

▼ Expand(All)
user1, user1_businesses, user2
(user2)-[UNNAMED188:Review]->(UNNAMED200)
3,040,043 estimated rows
8,321,478 db hits

6,684,446 rows

▼ Filter
user1, user1_businesses, user2
`anon[200]`:ReviewNode
3,040,043 estimated rows

Started streaming 16061 records after 50 ms and completed after 68 ms

优化:

首先, 可以为userid和businessid建立索引

CREATE INDEX FOR (user:UserNode {userid: 'tvZKPah2u9G9dFBg5GT0eg'})-[:Review]->(:BusinessNode)

CREATE INDEX FOR (b:BusinessNode) ON (b.businessid)

Added 1 index, completed after 26 ms.

Added 1 index, completed after 4 ms.

然后将上面的COLLECT函数改为仅对business.businessid聚合, 减少内存开销

MATCH (user1:UserNode {userid: 'tvZKPah2u9G9dFBg5GT0eg'})-[:Review]->(:ReviewNode)-[:Reviewed]->(b:BusinessNode)

WITH user1, COLLECT(DISTINCT b.businessid) AS user1_businesses

MATCH (user2:UserNode)-[:Review]->(:ReviewNode)-[:Reviewed]->(b:BusinessNode)

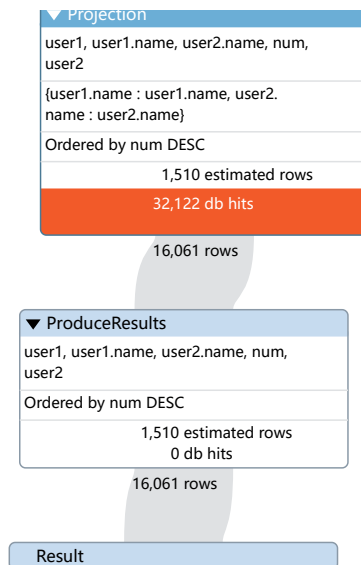
WHERE NOT EXISTS ((user1)-[:HasFriend]->(user2)) AND b.businessid IN user1_businesses AND user1<>user2

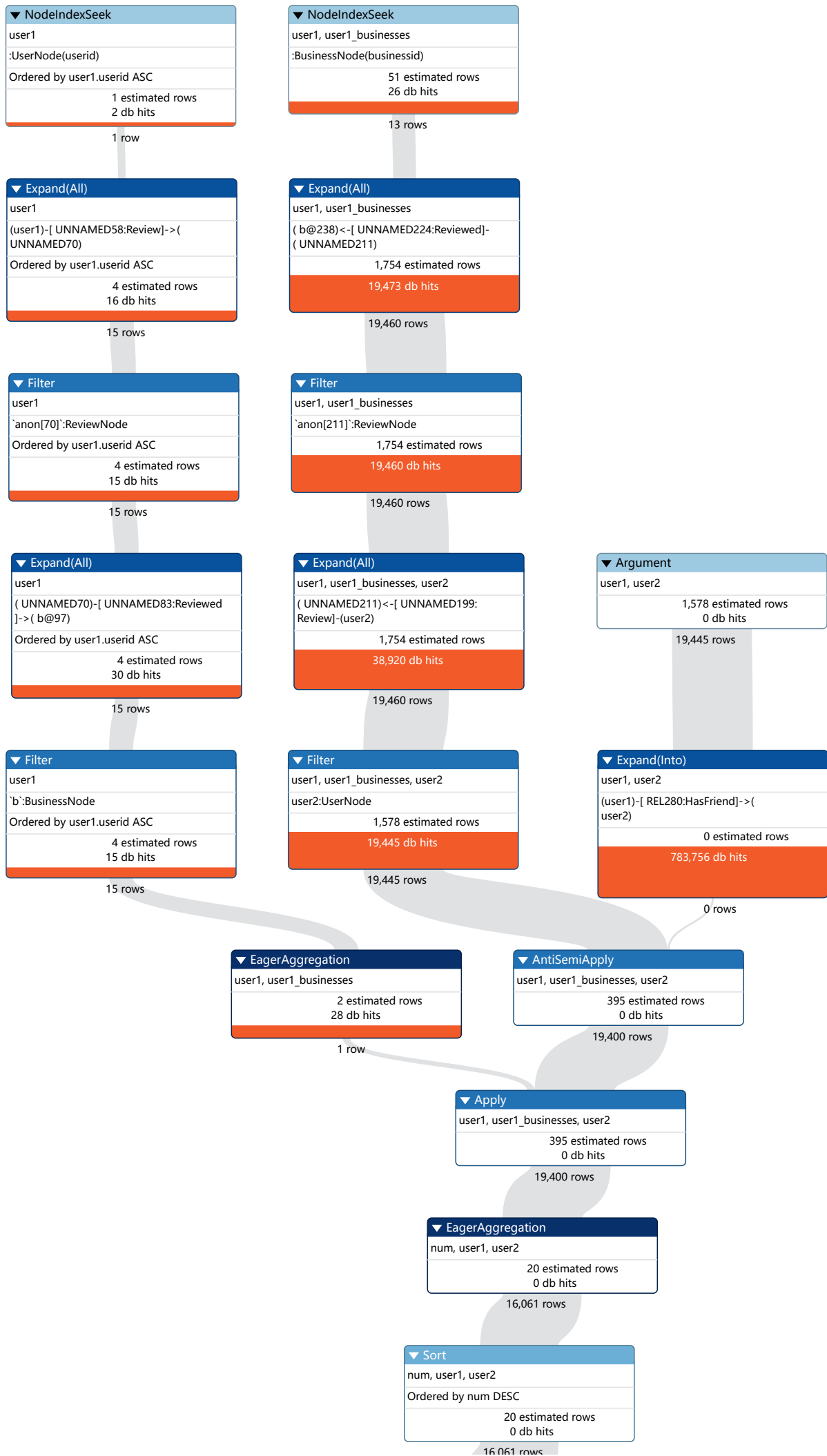
WITH user1, user2, COUNT(DISTINCT b) AS num

RETURN user1.name, user2.name, num

ORDER BY num DESC

Started streaming 4904 records after 1 ms and completed after 1 ms, displaying first 1000 rows.







18.


Neo4j查询:


```
MATCH (:ReviewNode {reviewid:'TIYgnDzezfeEnVeu9jHeEw'})[:Reviewed]->
(business:BusinessNode)
RETURN business
```

Projection
user1, user1.name, user2.name, num, user2
(user1.name : user1.name, user2.name : user2.name)
Ordered by num DESC
20 estimated rows
32,122 db hits

Graph

Table

Text

Code

business

```
{
  "identity": 8339259,
  "labels": [
    "BusinessNode"
  ],
  "properties": {
    "reviewcount": "39",
    "address": "405 Rue Sherbrooke Est",
    "city": "Montréal",
    "latitude": "45.517348",
    "businessid": "I3UkP4Mmp0cmfe3vTev0jw",
    "name": "Sushi 999",
    "stars": "2.5",
    "longitude": "-73.5677004"
  }
}
```

Started streaming 1 records after 1 ms and completed after 18146 ms.

MongoDB查询:

```
var bid=db.review.findOne({"review_id":"'TIYgnDzezfeEnVeu9jHeEw'").business_id
db.business.findOne({"business_id":bid})
```

```

> var bid=db.review.findOne({"review_id":"TIYgnDzezfeEnVeu9jHeEw"}).business_id
> db.business.findOne({"business_id":bid})
{
  "_id" : ObjectId("6016c6b5af81085b0f21b190"),
  "business_id" : "I3UkP4Mmp0cmfe3vTev0jw",
  "name" : "Sushi 999",
  "address" : "405 Rue Sherbrooke Est",
  "city" : "Montréal",
  "state" : "QC",
  "postal_code" : "H2L 1J9",
  "latitude" : 45.517348,
  "longitude" : -73.5677004,
  "stars" : 2.5,
  "review_count" : 39,
  "is_open" : 0,
  "attributes" : {
    "RestaurantsGoodForGroups" : "True",
    "BikeParking" : "False",
    "NoiseLevel" : "u'average'",
    "GoodForKids" : "True",
    "WiFi" : "u'free'",
    "BusinessParking" : "{ 'garage': False, 'street': True, 'validated': False,
'lot': False, 'valet': False}",
    "RestaurantsDelivery" : "True",
    "RestaurantsReservations" : "True",
    "Alcohol" : "'beer_and_wine'",
    "HasTV" : "True",
    "RestaurantsPriceRange2" : "2",
    "OutdoorSeating" : "False",
    "RestaurantsTakeOut" : "True",
    "RestaurantsAttire" : "u'casual'",
    "Ambience" : "{ 'romantic': False, 'intimate': False, 'classy': False, 'hip
ster': False, 'divey': False, 'touristy': False, 'trendy': False, 'upscale': False, 'casua
l': True}"
  },
  "categories" : [
    "Sushi Bars",
    "Japanese",
    "Restaurants"
  ],
  "hours" : null,
  "loc" : {
    "type" : "Point",
    "coordinates" : [
      -73.5677004,
      45.517348
    ]
  }
}

```

MongoDB查询所用时间略大于Neo4j，据此可以指出Neo4j和MongoDB主要的适用场景。

Neo4j适用于复杂的图结构数据查询，如社交网络、推荐系统、知识图谱等。由于其使用基于图论的查询语言Cypher，支持关系型数据的快速查询和分析。

而MongoDB适用于海量非结构化或半结构化数据存储和查询，如日志、传感器数据、文档数据库等。由于其使用文档数据库模型，支持高效的数据插入和查询，以及分布式数据库集群的横向扩展。

Lab 4

1.

```

MATCH (user:UserNode)-[:Review]->(:ReviewNode)-[:Reviewed]->(business:BusinessNode)
WITH user, COUNT(distinct business) as count
WHERE count>5
RETURN user.name, user.funny, user.fans, count

```

224936	Crystal	0	0		
224937					
224938					

2.

将1得到的结果导入MongoDB，并使用该表格数据，统计其中所有出现的用户名及该用户名对应的出现次数，并按照出现次数降序排序,使用aggregate实现

- 1) 从Neo4j的查询中导出csv文件 (export.csv)
- 2) 在mongodb新建集合from_neo4j，将csv文件导入集合

C:\GAP\大数据管理实验>scp ./export.csv root@1.94.55.43:/root/

```
C:\GAP\大数据管理实验>scp ./export.csv root@1.94.55.43:/root/
The authenticity of host '1.94.55.43 (1.94.55.43)' can't be established.
ED25519 key fingerprint is SHA256:H04FMGJ9ay3Di2cnNBp/IZ3bkotVikwwN8BWJLP4NR8.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Warning: Permanently added '1.94.55.43' (ED25519) to the list of known hosts.
root@1.94.55.43's password:
export.csv 100% 2375KB 8.7MB/s 00:00
```

```
root@ecs-c925:~# ls
data export.csv mysql-apt-config_0.8.10-1_all.deb
```

然后启动mongo，选择yelp数据库，创建一个新的集合from_neo4j

```
db.createCollection("from_neo4j")
```

```
> db.createCollection("from_neo4j")
{ "ok" : 1 }
```

使用show collections查看当前集合：

```
> show collections
Average_Stars
Subreview
business
from_neo4j
review
test_map_reduce
user
```

退出mongoDB，回到主目录，把数据导入到mongoDB中yelp数据集的from_neo4j集合中。

```
mongoimport -d=yelp -c=from_neo4j --type=csv --headerline ./export.csv
```

```
root@ecs-c925:~# mongoimport -d=yelp -c=from_neo4j --type=csv --headerline ./export.csv
2023-10-27T11:35:41.800+0800 connected to: mongodb://localhost/
2023-10-27T11:35:43.261+0800 224935 document(s) imported successfully. 0 document(s) failed to import.
```

- 3) 统计其中所有出现的用户名及该用户名对应的出现次数，并按照出现次数降序排序。

```
db.from_neo4j.aggregate([
{$group:{_id:'$u.name', count:{$sum:1}},
{$sort:{count: -1}}
])
```

```
{ "_id" : "John", "count" : 1864 }
{ "_id" : "Michael", "count" : 1804 }
{ "_id" : "David", "count" : 1736 }
{ "_id" : "Chris", "count" : 1687 }
{ "_id" : "Jennifer", "count" : 1649 }
{ "_id" : "Mike", "count" : 1571 }
{ "_id" : "Jessica", "count" : 1469 }
{ "_id" : "Sarah", "count" : 1346 }
{ "_id" : "Michelle", "count" : 1333 }
{ "_id" : "Lisa", "count" : 1204 }
{ "_id" : "Jason", "count" : 1101 }
{ "_id" : "Mark", "count" : 1088 }
{ "_id" : "Ashley", "count" : 1083 }
{ "_id" : "Amy", "count" : 1017 }
{ "_id" : "Amanda", "count" : 1013 }
{ "_id" : "Stephanie", "count" : 1003 }
{ "_id" : "Brian", "count" : 993 }
{ "_id" : "J", "count" : 986 }
{ "_id" : "Melissa", "count" : 953 }
{ "_id" : "Nicole", "count" : 943 }
Type "it" for more
```

3.

MATCH (business:BusinessNode)-[:IN_CATEGORY]->(c:CategoryNode)

RETURN business.name as name, business.city as city, c.category as category

	name	city	category
1	"Rohaley's Auto & Truck Repair"	"Mentor"	"Commercial Truck Repair"
2	"AAA Muffler"	"Henderson"	"Commercial Truck Repair"
3	"Parkway Auto Care"	"Berea"	"Commercial Truck Repair"
4	"Master Tech Collision"	"Tempe"	"Commercial Truck Repair"
5	"Regal Truck & Trailer Repair"	"Mississauga"	"Commercial Truck Repair"
6	"Aone Mobile Mechanics"	"Las Vegas"	"Commercial Truck Repair"
7	"Diversified Truck & Equipment Sales Inc"	"Mesa"	"Commercial Truck Repair"
8	"Sizzling Hut"	"DeForest"	"Szechuan"
9	"Spicy City"	"Las Vegas"	"Szechuan"
10	"China Star"	"Pittsburgh"	"Szechuan"
11	"Tasty China"	"North Las Vegas"	"Szechuan"

Started streaming 788359 records after 1 ms and completed after 3 ms, displaying first 1000 rows.

然后类似于上面第2题的步骤:

```
root@ecs-c925:~# ls
AllBusiness.csv data
```

db.createCollection("AllBusiness")

```
> db.createCollection("AllBusiness")
{ "ok" : 1 }
```

退出mongoDB，回到主目录，把数据导入到mongoDB中yelp数据集的AllBusiness集合中。

```
mongoimport -d=yelp -c=AllBusiness --type=csv --headerline ./AllBusiness.csv
```

```
root@ecs-c925:~# mongoimport -d=yelp -c=AllBusiness --type=csv --headerline ./AllBusiness.csv
2023-10-27T15:40:02.973+0800    connected to: mongod://localhost/
2023-10-27T15:40:05.973+0800    [#####.....] yelp.AllBusiness      16.1MB/32.3MB (50.0%)
2023-10-27T15:40:08.973+0800    [#####.] yelp.AllBusiness      31.7MB/32.3MB (98.0%)
2023-10-27T15:40:09.090+0800    [#####.] yelp.AllBusiness      32.3MB/32.3MB (100.0%)
2023-10-27T15:40:09.090+0800    788359 document(s) imported successfully. 0 document(s) failed to import.
```

接下来使用aggregate对AllBusiness去重，仅保留城市、商铺类型。首先创建一个集合用于保存结果

```
db.createCollection("DistinctBusiness")
```

```
db.AllBusiness.aggregate([{$group: { id: { city: '$city', category: '$category' } } }]).forEach((item) => {
db.DistinctBusiness.insert( item.id ) })
```

查看结果：

```
> db.DistinctBusiness.count()
67536
> db.DistinctBusiness.find().limit(5)
{ "_id" : ObjectId("653b6a55c84e3a41a5adf21b"), "city" : "Brecksville", "category" : "Fitness & Instruction" }
{ "_id" : ObjectId("653b6a55c84e3a41a5adf21c"), "city" : "Phoenix", "category" : "Elementary Schools" }
{ "_id" : ObjectId("653b6a55c84e3a41a5adf21d"), "city" : "Oberlin", "category" : "Pets" }
{ "_id" : ObjectId("653b6a55c84e3a41a5adf21e"), "city" : "Goodyear", "category" : "Driving Schools" }
{ "_id" : ObjectId("653b6a55c84e3a41a5adf21f"), "city" : "Markham", "category" : "Building Supplies" }
```

将结果导出到服务器主目录下的result.csv中。

```
mongoexport -d yelp -c DistinctBusiness --type=csv --fields city,category --out result.csv
```

然后将其放在neo4j安装目录的import下

```
cd ~/neo4j-community-4.0.9/import
```

```
cp /root/result.csv ./
```

将去重后的结果导入Neo4j中的新库result中，完成（City-[Has]->Category）图谱的构建。

```
LOAD CSV WITH HEADERS FROM "file:///result.csv" AS f
MERGE (c:CityNode {city: COALESCE(f.city, "")})
MERGE (a:CategoryNode {category: COALESCE(f.category, "")})
CREATE (c) -[:Has]-> (a)
```

Added 125 labels, created 125 nodes, set 125 properties, created 67536 relationships, completed after 234695 ms.

最后查看City-[Has]->Category图谱

```
MATCH p=()-[:Has]->()
```

```
RETURN p
```

```
LIMIT 20
```



Lab 5

1.

体验 MySQL 在 InnoDB 存储引擎下的 MVCC 多版本并发控制，实现的事务 ACID 特性。请注意 MySQL 需要选用什么事务隔离级来支持 MVCC？请构造多用户多写多读案例来展现 MVCC 并发控制特性，解释各种结果产生的原因。

在MySQL中，首先创建一个数据库testdb

```
mysql -u root -p
```

```
create database testdb;
```

```
use testdb;
```

```
mysql> create database testdb;
Query OK, 1 row affected (0.00 sec)

mysql> use testdb;
Database changed
```

在testdb中创建一个新表，设置engine=InnoDB

```
create table test (
  id int(10) not null,
  name varchar(20) not null,
  flag int(5) not null,
  primary key(id)
) engine=InnoDB;
```

设置事务隔离等级为可重复读 (repeatable read)

```
set session transaction isolation level repeatable read;
```

插入初始数据:

```
insert into test VALUES(1, 'LJP', 27);
```

```
insert into test VALUES(2, 'DJE', 42);
```

```
insert into test VALUES(3, 'OFW', 61);
```

```
insert into test VALUES(4, 'SLX', 15);
```

使用select * from test;查看当前表内的数据:

```
mysql> select * from test
-> ;
+----+-----+-----+
| id | name | flag |
+----+-----+-----+
| 1  | LJP  | 27   |
| 2  | DJE  | 42   |
| 3  | OFW  | 61   |
| 4  | SLX  | 15   |
+----+-----+-----+
4 rows in set (0.00 sec)
```

开始一个事务:

```
start transaction;
```

然后打开会话窗口B（直接在Xshell中完成）。窗口A将id为3的flag更新为88，与此同时，在另外一边，窗口B将id为3的flag更新为99。

```
start transaction;
```

```
update test set flag=88 where id=3; //窗口A
```

```
update test set flag=99 where id=3; //窗口B
```

在窗口A中查询当前表内的数据，可以发现A对id为3的数据更新成功，在本地查询的flag为88。但是对于窗口B，其对id为3的数据的更新操作受到阻塞（“Query execution was interrupted”），且在二次查询时发现该数据的flag仍然为A更新之前的值，也就是说在窗口A对数据的更新操作提交（commit）之前，会屏蔽窗口B对同一数据的更新操作，同时由于MVCC的特性，窗口B在查询数据时仍然会使用本地保存的版本。

现在，再在窗口A中完成提交commit操作，到窗口B中查询当前数据，发现此时的数据变为最新版本。此时窗口B可以正常对id为3的数据进行更新:

2.

体验 MongoDB 的 MVCC，数据集可自建或选用 yelp 数据集集中的 test 集合中进行测试，测试方法同 MySQL。请对测试结果进行说明，并与 MySQL 的 MVCC 实验结果进行对比分析。建议创建 MongoDB 副本或分片集群，体验 MVCC 的不同效果（可选做其一）。

名称/ID	监控	安全	状态	可用区	规格/镜像	IP地址	计费模式	标签	操作
<input type="checkbox"/> ecs-c925 17ee4adc-bd8c-441d-...			关机	可用区3	2vCPUs 4GiB c7.large.2 Ubuntu 16.04 server 64bit	60.204.242.167 (弹性公网) 5 Mbit/s 192.168.0.186 (私有)	按量计费	2023/10/30 ...	远程登录 更多
<input checked="" type="checkbox"/> test-0001 42b72203-104d-47e6-...			运行中	可用区3	2vCPUs 4GiB c7.large.2 Ubuntu 16.04 server 64bit	60.204.242.167 192.168.0.186	按量计费	2023/10/30 ...	远程登录 更多
<input type="checkbox"/> test-0002 8cd9fb68-84c4-43fc-a...			运行中	可用区3	2vCPUs 4GiB c7.large.2 Ubuntu 16.04 server 64bit	60.204.242.167 192.168.0.186	按量计费	2023/10/30 ...	远程登录 更多
<input type="checkbox"/> test-0003 2e5bcf52-a603-449a-8...			运行中	可用区3	2vCPUs 4GiB c7.large.2 Ubuntu 16.04 server 64bit	60.204.242.167 192.168.0.186	按量计费	2023/10/30 ...	远程登录 更多

创建三台服务器(test-0001 test-0002 test-0003), 弹性公网ip分别为:

60.204.242.167

60.204.228.189

60.204.244.116

添加入方向和出方向规则, 检验是否可以ping通:

```
Microsoft Windows [版本 10.0.22621.2428]
(c) Microsoft Corporation。保留所有权利。

C:\Users\xiaoy>ping 60.204.228.189

正在 Ping 60.204.228.189 具有 32 字节的数据:
来自 60.204.228.189 的回复: 字节=32 时间=65ms TTL=46
来自 60.204.228.189 的回复: 字节=32 时间=52ms TTL=46
来自 60.204.228.189 的回复: 字节=32 时间=52ms TTL=46
来自 60.204.228.189 的回复: 字节=32 时间=52ms TTL=46

60.204.228.189 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 52ms, 最长 = 65ms, 平均 = 55ms
```

三台服务器分别创建数据存放的目录:

```
root@test-0001:~# su - mongod
No directory, logging in with HOME=/
root@test-0001:~# mkdir /usr/local/mongod
root@test-0001:~# cd /usr/local/mongod
root@test-0001:/usr/local/mongod# mkdir -p data/shard11
root@test-0001:/usr/local/mongod# mkdir -p data/shard21
root@test-0001:/usr/local/mongod# mkdir -p data/config
root@test-0001:/usr/local/mongod# touch data/shard11.log
root@test-0001:/usr/local/mongod# touch data/shard21.log
root@test-0001:/usr/local/mongod#
```

```

root@test-0002:~# su - mongod
No directory, logging in with HOME=/
root@test-0002:~# mkdir /usr/local/mongod
root@test-0002:~# cd /usr/local/mongod
root@test-0002:/usr/local/mongod# mkdir -p data/shard12
root@test-0002:/usr/local/mongod# mkdir -p data/shard22
root@test-0002:/usr/local/mongod# mkdir -p data/config
root@test-0002:/usr/local/mongod# touch data/shard12.log
root@test-0002:/usr/local/mongod# touch data/shard22.log
root@test-0002:/usr/local/mongod# █

```

```

root@test-0003:~# su - mongod
No directory, logging in with HOME=/
root@test-0003:~# mkdir /usr/local/mongod
root@test-0003:~# cd /usr/local/mongod
root@test-0003:/usr/local/mongod# mkdir -p data/shard13
root@test-0003:/usr/local/mongod# mkdir -p data/shard23
root@test-0003:/usr/local/mongod# mkdir -p data/config
root@test-0003:/usr/local/mongod# touch data/shard13.log
root@test-0003:/usr/local/mongod# touch data/shard23.log
root@test-0003:/usr/local/mongod# █

```

分别完成shard1和shard2的replica set配置:

```

root@test-0001:/usr/local/mongod# mongod --shardsvr --replSet shard1 --port 27017 --dbpath /usr/local/mongod/data/shard11 --oplogSize 2048 --logpath /usr/local/mongod/data/shard11.log --logappend --bind_ip=0.0.0.0 --fork
about to fork child process, waiting until server is ready for connections.
forked process: 8502
child process started successfully, parent exiting

root@test-0001:/usr/local/mongod# mongod --shardsvr --replSet shard2 --port 27018 --dbpath /usr/local/mongod/data/shard21 --oplogSize 2048 --logpath /usr/local/mongod/data/shard21.log --logappend --bind_ip=0.0.0.0 --fork
about to fork child process, waiting until server is ready for connections.
forked process: 8612
child process started successfully, parent exiting

```

初始化shard1和shard2的replica set:

shard1, 在某一台服务器上执行:

mongo 60.204.242.167:27017

```

config = { id: 'shard1', members: [ {id: 0, host: '60.204.242.167:27017'}, {id: 1, host: '60.204.228.189:27017'}, {id: 2, host: '60.204.244.116:27017'} ] }

```

rs.initiate(config);

shard2, 在某一台服务器上执行:

mongo 60.204.242.167:27018

```

config = { id: 'shard2', members: [ {id: 0, host: '60.204.242.167:27018'}, {id: 1, host: '60.204.228.189:27018'}, {id: 2, host: '60.204.244.116:27018'} ] }

```

rs.initiate(config);

```

root@test-0001:/usr/local/mongodb# mongo 60.204.242.167:27017
MongoDB shell version v4.4.17
connecting to: mongodb://60.204.242.167:27017/test?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("2498494b-f982-4229-98f7-fcfaed2591d6") }
MongoDB server version: 4.4.17
...
The server generated these startup warnings when booting:
  2023-10-30T20:31:52.990+08:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
  2023-10-30T20:31:53.632+08:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
  2023-10-30T20:31:53.632+08:00: You are running this process as the root user, which is not recommended
...
...
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
  metrics about your deployment (disk utilization, CPU, operation statistics, etc).

  The monitoring data will be available on a MongoDB website with a unique URL accessible to you
  and anyone you share the URL with. MongoDB may use this information to make product
  improvements and to suggest MongoDB products and deployment options to you.

  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
...
> config = { _id: 'shard1', members: [ { _id: 0, host: '60.204.242.167:27017' }, { _id: 1, host: '60.204.228.189:27017' }, { _id: 2, host: '60.204.244.116:27017' } ] }
{
  "_id" : "shard1",
  "members" : [
    {
      "_id" : 0,
      "host" : "60.204.242.167:27017"
    },
    {
      "_id" : 1,
      "host" : "60.204.228.189:27017"
    },
    {
      "_id" : 2,
      "host" : "60.204.244.116:27017"
    }
  ]
}
> rs.initiate(config);
{ "ok" : 1 }
shard1:SECONDARY>

```

```

> config = { _id: 'shard2', members: [ { _id: 0, host: '60.204.242.167:27018' }, { _id: 1, host: '60.204.228.189:27018' }, { _id: 2, host: '60.204.244.116:27018' } ] }
{
  "_id" : "shard2",
  "members" : [
    {
      "_id" : 0,
      "host" : "60.204.242.167:27018"
    },
    {
      "_id" : 1,
      "host" : "60.204.228.189:27018"
    },
    {
      "_id" : 2,
      "host" : "60.204.244.116:27018"
    }
  ]
}
> rs.initiate(config);
{
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1698669536, 1),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"),
      "keyId" : NumberLong(0)
    }
  },
  "operationTime" : Timestamp(1698669536, 1)
}
shard2:SECONDARY>

```

配置config server:

在三台服务器上分别执行:

```

mongod --configsvr --replSet config --dbpath /usr/local/mongodb/data/config --port 20000 --
logpath /usr/local/mongodb/data/config.log --logappend --bind_ip=0.0.0.0 --fork

```

在某一台服务器上执行:

```

mongo 60.204.242.167:20000

```

```

config = { id: 'config', members: [ {id: 0, host: '60.204.242.167:20000'}, {id: 1, host:
'60.204.228.189:20000'}, {id: 2, host: '60.204.244.116:20000'}] }

```

```

rs.initiate(config);

```

```
config:SECONDARY>
```

配置mongos:

在三台服务器上分别执行：

```
mongos --configdb config/60.204.242.167:20000,60.204.228.189:20000,60.204.244.116:20000 --  
port 30000 --logpath /usr/local/mongodb/data/mongos.log --logappend --bind_ip=0.0.0.0 --fork
```

使用mongos:

mongo 60.204.242.167:30000

切换到admin，添加分片：

```
use admin;
```

```
db.runCommand({addshard:"shard1/60.204.242.167:27017,60.204.228.189:27017,60.204.244.116:27017",name:"s1", maxsize:20480});
```

```
db.runCommand({addshard:"shard2/60.204.242.167:27018,60.204.228.189:27018,60.204.244.116:27018",name:"s2", maxsize:20480});
```

(上图为报错，因为addshard字段内严格不允许出现空格)

```

root@test-0001:/usr/local/mongodb# mongo 60.204.242.167:30000
MongoDB shell version v4.4.17
connecting to: mongodb://60.204.242.167:30000/test?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("d7add2e-c143-4e61-87f5-45ff525820b8") }
MongoDB server version: 4.4.17
...
The server generated these startup warnings when booting:
  2023-10-30T20:44:18.813+08:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
  2023-10-30T20:44:18.813+08:00: You are running this process as the root user, which is not recommended
...
mongos> use admin;
switched to db admin
mongos> db.runCommand({addshard:"shard1/60.204.242.167:27017,60.204.228.189:27017,60.204.244.116:27017",name:"s1", maxsize:20480});
{
  "shardAdded" : "s1",
  "ok" : 1,
  "operationTime" : Timestamp(1698669951, 5),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1698669951, 5),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
mongos> db.runCommand({addshard:"shard2/60.204.242.167:27018,60.204.228.189:27018,60.204.244.116:27018",name:"s2", maxsize:20480});
{
  "shardAdded" : "s2",
  "ok" : 1,
  "operationTime" : Timestamp(1698669965, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1698669965, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
}
mongos>

```

激活数据库分片：

创建一个名为testdb的数据库：

use testdb

```

mongos> use testdb
switched to db testdb

```

激活分片：

sh.enableSharding("testdb")

```

mongos> sh.enableSharding("testdb")
{
  "ok" : 1,
  "operationTime" : Timestamp(1698670042, 8),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1698670042, 8),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
mongos>

```

使用sh.status()查看数据库当前情况：

```

mongos> sh.status()
--- Sharding Status ---
  sharding version: {
    "_id" : 1,
    "minCompatibleVersion" : 5,
    "currentVersion" : 6,
    "clusterId" : ObjectId("653fa4d673838d5e44d4bc05")
  }
  shards:
    { "_id" : "s1", "host" : "shard1/60.204.228.189:27017,60.204.242.167:27017,60.204.244.116:27017", "state" : 1 }
    { "_id" : "s2", "host" : "shard2/60.204.228.189:27018,60.204.242.167:27018,60.204.244.116:27018", "state" : 1 }
  active mongoses:
    "4.4.17" : 3
  autosplit:
    Currently enabled: yes
  balancer:
    Currently enabled: yes
    Currently running: no
    Failed balancer rounds in last 5 attempts: 0
    Migration Results for the last 24 hours:
      28 : Success
  databases:
    { "_id" : "config", "primary" : "config", "partitioned" : true }
      config.system.sessions
        shard key: { "_id" : 1 }
        unique: false
        balancing: true
        chunks:
          s1      996
          s2      28
    too many chunks to print, use verbose if you want to force print
    { "_id" : "testdb", "primary" : "s2", "partitioned" : true, "version" : { "uuid" : UUID("1b3b503d-a2b2-4e9c-b435-be52f5d5c4dc"), "lastMod" : 1 } }
mongos>

```

创建一个新的集合testc:

```
db.createCollection("testc")
```

```
mongos> db.createCollection("testc")
{
  "ok" : 1,
  "operationTime" : Timestamp(1698673262, 1),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1698673262, 1),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
```

插入一些数据

```
db.testc.insertMany([
  {"id": 1, "name": "LJP", "age": 27},
  {"id": 2, "name": "DJE", "age": 42},
  {"id": 3, "name": "OFW", "age": 61},
  {"id": 4, "name": "SLX", "age": 15}
]);
```

```
mongos> db.testc.insertMany([
...   {"_id": 1, "name": "LJP", "age": 27},
...   {"_id": 2, "name": "DJE", "age": 42},
...   {"_id": 3, "name": "OFW", "age": 61},
...   {"_id": 4, "name": "SLX", "age": 15}
... ]);
{ "acknowledged" : true, "insertedIds" : [ 1, 2, 3, 4 ] }
mongos>
```

通过db.testc.find()可以查看当前集合中的数据:

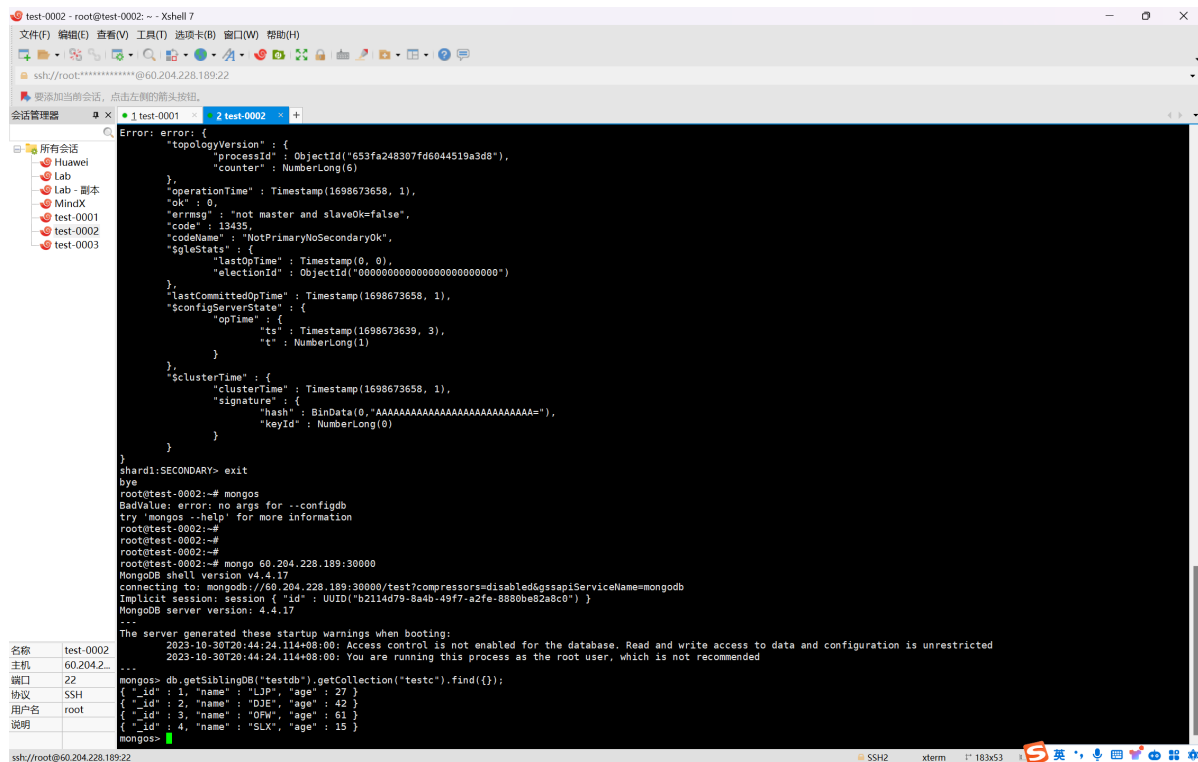
```
mongos> db.testc.find()
{ "_id" : 1, "name" : "LJP", "age" : 27 }
{ "_id" : 2, "name" : "DJE", "age" : 42 }
{ "_id" : 3, "name" : "OFW", "age" : 61 }
{ "_id" : 4, "name" : "SLX", "age" : 15 }
```

此时打开test-0002服务器进入mongos:

```
mongo 60.204.228.189:30000
```

查询数据:

```
db.getSiblingDB("testdb").getCollection("testc").find({});
```

删除一条数据：

```
db.getSiblingDB("testdb").getCollection("testc").deleteOne({ _id: 3 })
```

```
mongos> db.getSiblingDB("testdb").getCollection("testc").deleteOne({ _id:3 })
{ "acknowledged" : true, "deletedCount" : 1 }
```

再回到test-0001服务器，通过db.testc.find()查看当前集合中的数据：

```
mongos> db.testc.find()
{ "_id" : 1, "name" : "LJP", "age" : 27 }
{ "_id" : 2, "name" : "DJE", "age" : 42 }
{ "_id" : 4, "name" : "SLX", "age" : 15 }
```

MySQL和mongoDB的MVCC对比分析：

MySQL和MongoDB都实现了MVCC（多版本并发控制）机制，用以解决读写冲突的并发控制。在MVCC机制中，为事务分配单向增长的时间戳，并为每个修改保存一个版本，版本与事务时间戳关联。读操作只会读取该事务开始前的数据库快照，从而避免阻塞其它读操作。

虽然二者都采用了MVCC，但在具体实现上存在显著差异。MySQL的MVCC是通过保存数据的历史版本来实现的，每个数据项都有一个时间戳timestamp，可以追踪到创建和修改的时间点。这种实现方式使得MySQL能够提供严格的可重复读，保证读取操作不会看到未提交（commit）的修改，对已修改但未提交的数据进行更新也是非法的。

MongoDB的MVCC实现更复杂。在MongoDB中，每个文档都有一个包含多个版本的时间戳数组，也可以看作一个topologyVersion（拓扑版本号）。当文档被修改时，旧版本并不会被删除，而是存储在数组中，而在绝大部分情况下读写操作都使用的是当前最新的版本（当然，这种方式也允许MongoDB在读取数据时查看过时的数据版本），从而实现非阻塞读写操作。